BANCHE DATI PER TUTTI

Commodore COMPUTER CLUB

76 L. 6.000

La rivista degli ute

Anno IX - N. 76 Luglio/Agosto 1990

Sped. Abb. Post. Gr III/70 - CR -Distr.: Parrini

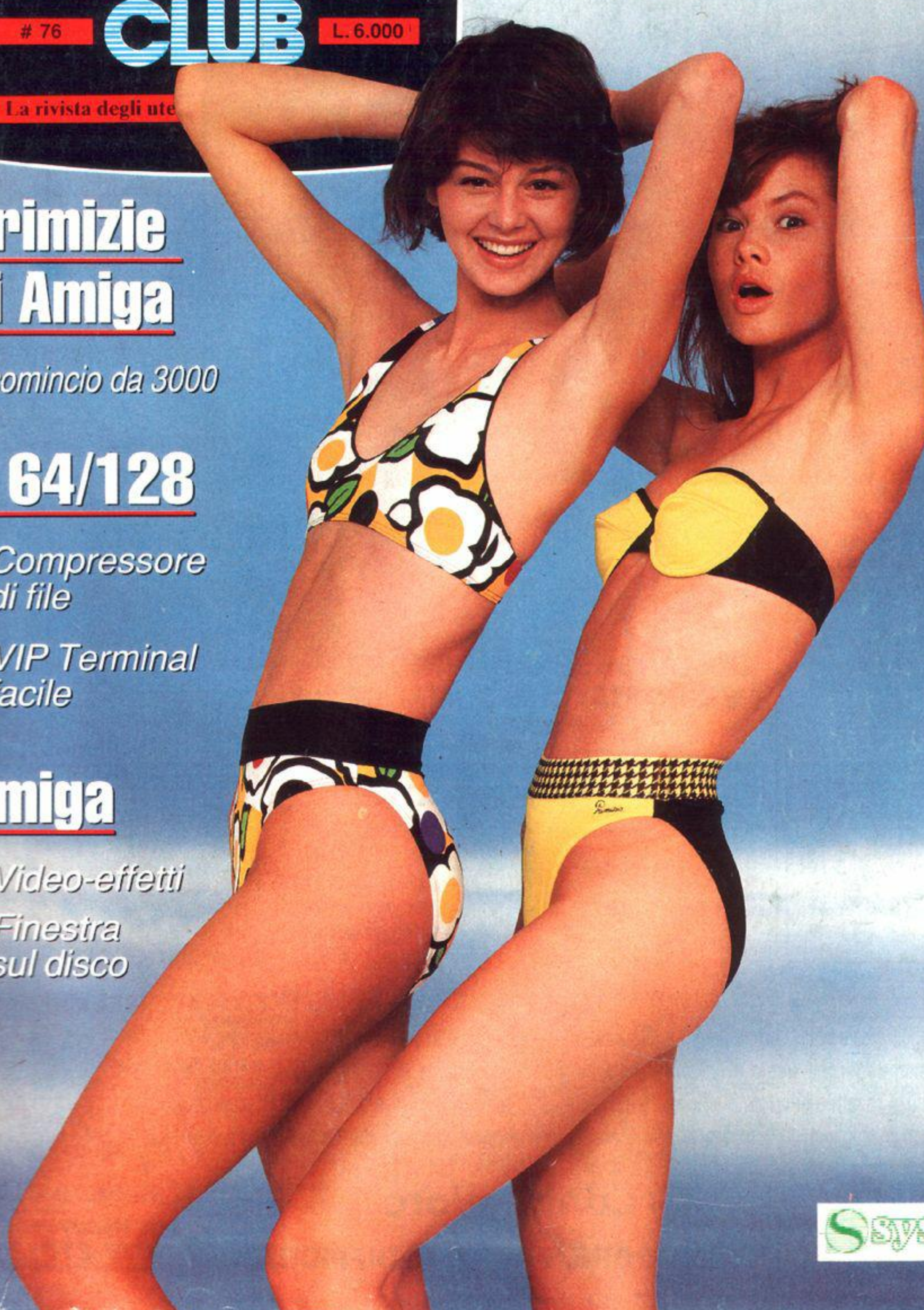
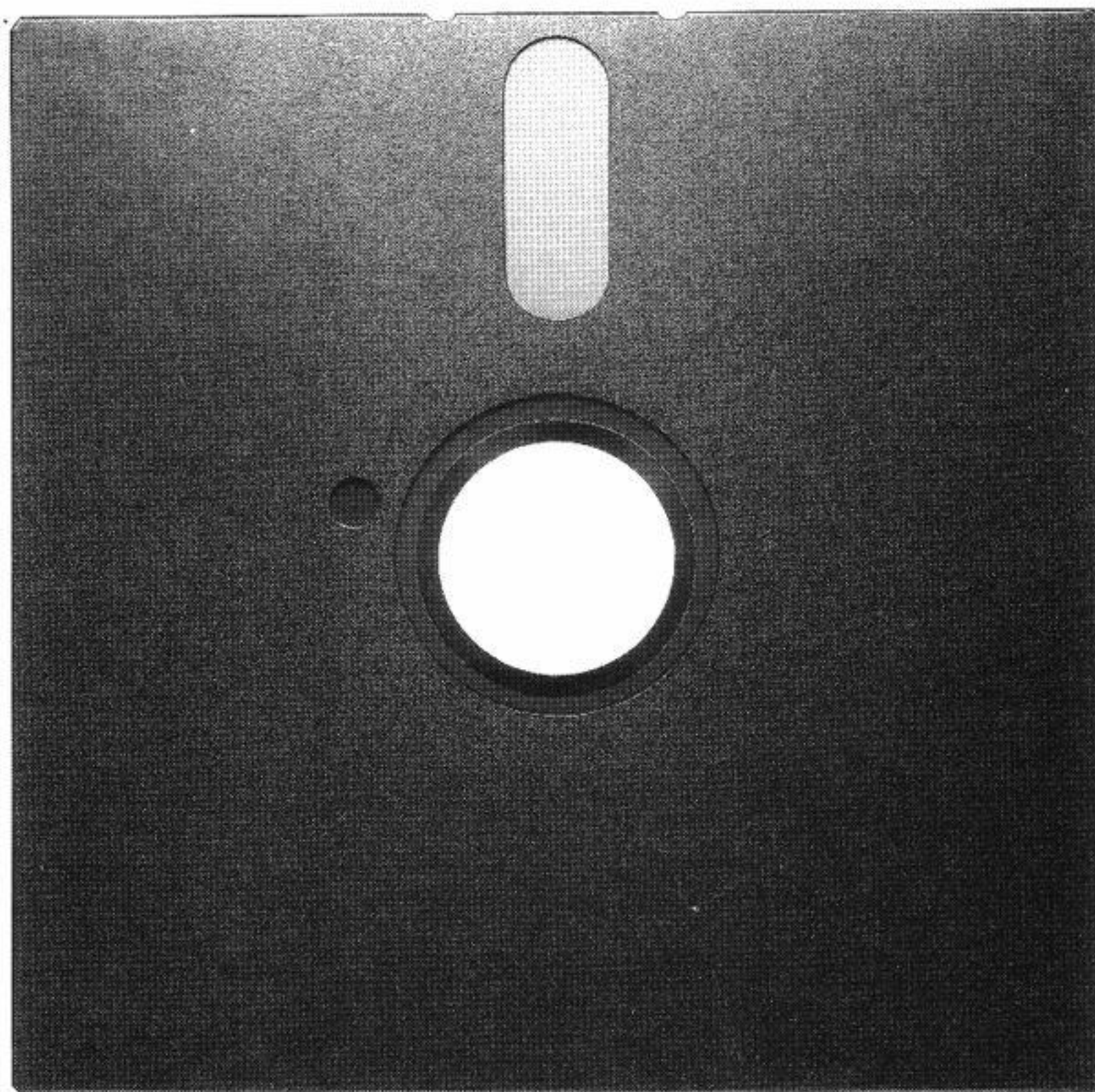
Primizie di Amiga

Ricomincio da 3000

C 64/128

- *Compressore di file*
- *VIP Terminal facile*

Amiga

- *Video-effetti*
- *Finestra sul disco*

DOMANDA: È COMPATIBILE?
RISPOSTA: CHE DOMANDE.

Da oggi, grazie ai PC Commodore, problemi tecnici e problemi pratici si risolvono più facilmente. Commodore Italiana, infatti, ha creato e garantisce in prima persona una linea di personal capace di rispondere alle esigenze di tutti e di lavorare e dialogare con tutti: dall'utente più sofisticato al neofita più acerbo. Da oggi, invece di scegliere un semplice PC, scegliete di fare un investimento garantito da Commodore Italiana.

Commodore

PC COMMODORE. FACILE IL DIFFICILE.

Sommario

Foto di copertina: PRIMIZIA di Visconte di Modrone

Da pagina **41** a pagina **56** la consueta rassegna dei videogames in arrivo; ed i relativi commenti, severi come al solito!

Campus 64 / 128

Usa il tuo computer

Rubriche

Forse non tutti sanno che...

Dal mese di **luglio '90** verranno limitate le collaborazioni per il C/128.

Dal **gennaio '91** verrà evasa prevalentemente la corrispondenza pervenuta a mezzo BBS (modem).

Dal **gennaio '91** verranno privilegiate le collaborazioni che perverranno in Redazione a mezzo BBS.

Dal **gennaio '91** non verranno più affrontati argomenti relativi al C/128.

Dal **dicembre '91** non verranno più affrontati argomenti relativi al C/64.

Commodore Computer Club

COMMODORE COMPUTER CLUB

Direttore: Alessandro de Simone
Assistente di redazione: Marco Miotti

Redazione / Collaboratori:
Davide Ardizzone - Claudio Baiocchi
Luigi Callegari - Umbero Colapicchioni
Donato De Luca - Carlo D'Ippolito
Valerio Ferri - Michele Maggi
Giancarlo Mariani - Domenico Pavone
Armando Sforzi - Dario Pistella
Fabio Sorgato- Valentino Spataro
Franco Rodella- Stefano Simonelli
Luca Viola

Grafica: Arturo Ciaglia

Redazione:
Via Mosè, 22 cap. 20090 OPERA (Mi)

Telefoni 02 / 57.60.63.10
Fax 02 / 57.60.30.39
BBS 02 / 52.49.211

Pubblicità:
Leandro Nencioni (dir. vendite)
Ketty Cusin
Via Mose', 22 20090 Opera (Mi)
tel. 02 / 57.60.63.10
Spazio Nuovo
Via P. Foscari, 70
cap. 00139 Roma
tel. 06 / 81.09.679

Edizioni:
Systems Editoriale s.r.l.
Via Mose, 22 20090 Opera (MI)
Reg. Naz. Stampa N. 01500
vol. 15 fg. 793

Abbonamenti: Liliana Spina
Arretrati e s/w: Lucia Dominoni
Tel. 02 / 57.60.63.10

Tariffe: Prezzo per copia L. 6000
Abbonamento annuo (11 fascicoli) L. 60000
Estero: L. 100000 - Indirizzare versamenti a:
Systems Editoriale Srl c/c 37952207 oppure
inviare comune assegno bancario non
trasferibile e barrato due volte a:
Systems Editoriale Srl (servizio arretrati)
Via Mose', 22
cap. 20090 OPERA (Mi)

Composizione e fotolito:
Systems Editoriale
Stampa: La Litografica Srl Guggiono (MI)

Registrazioni: Tribunale di Milano
n. 370 del 2/10/82

Direttore Responsabile: Michele Di Pisa

Spedizioni in abbonamento postale gruppo
III. Pubblicita' inferiore al 70%

Distributore: Parrini - Milano

Pubblicazioni Systems:
Banca Oggi
Commodore Club (disco)
Commodore Computer Club
Commodore Computer Club
(disco, edizione tedesca)
Computer quotidiano
Electronic Mass Media Age
Hospital Management - Nursing '90
Personal Computer
Jonathan - TuttoGatto
Videoteca - VR Videoregistrare

Editoriale

Prima di andare in vacanza ho tolto, dalla scrivania sulla quale giaceva da quasi quattro anni, il "vecchio" **C/64** nella sua configurazione completa: speed dos, drive 1541 velocizzato, drive 1541 "normale", registratore, monitor a colori, amplificatore esterno, joystick, mouse e stampante.

Ho invece lasciato, sulla scrivania adiacente, il **C/128-D** (anch'esso in configurazione "da parata") che però vede limitato il suo spazio vitale dall'**Amiga** e dall'**Ms-Dos** 80286; questi, ormai, la fanno da padrone da quasi due anni.

Dopo aver riposto in cantina l'ingombrante fardello, che ora tiene compagnia agli impolverati **Vic-20** e **C/16**, mi è venuta la curiosità di sommare i soldi spesi per tutte quelle attrezzature. Roba da pazzi! Una cifra di sette caratteri (ed il primo, purtroppo, non era un "uno") è comparsa sul visore della calcolatrice tascabile. Ne sarà, almeno, valsa la pena?

Domenico Pavone confessa di pagare una bolletta SIP di importo più che doppio, rispetto al normale, da quando ha installato il modem. La sua decisione, però, non è quella di limitare l'uso telematico del proprio telefono, ma di procurarsi al più presto un modem da 19200 baud, per effettuare collegamenti transoceanici a costi ragionevoli.

Anch'io, del resto, tra breve sostituirò la mia 24 aghi con una laser ed il C/128 cederà il posto ad un **386** oppure all'**Amiga 3000**. Ripeto la domanda: ne vale la pena?

Per noi della Redazione la risposta è più che scontata; rimane da chiedersi se la stessa risposta è valida anche per il nostro lettore "medio".

E' piuttosto difficile, ovviamente, cercare termini di paragone, però tentiamo lo stesso. Supponiamo, ad esempio, che sia possibile ricavare 500 mila lire dalla vendita del vecchio sistema (C/64 + drive + stampante) che ci è costato, ai tempi, un milione.

Se ci siamo divertiti per un paio di anni con il C/64, potremmo considerare la "perdita" (500 mila lire) come un canone di locazione. In questo caso la cifra che ne risulta, circa 20000 lire al mese, non può certo esser considerata elevata.

Oppure possiamo paragonare la stessa cifra al numero di spettacoli cinematografici o teatrali cui è possibile assistere (un paio al mese).

Oppure al numero di giorni di vacanza (il tema è più che attuale...) che è possibile trascorrere in alta stagione: ma, al giorno d'oggi, che vacanza potete fare con 500 mila lirette?

Comunque girate la faccenda, però, rimane il fatto che la vecchia tastiera, ormai, manifesta segni di stanchezza.

E non venitemi a dire che non ve ne siete ancora accorti...

A. d.S.

Arkanoid infinito

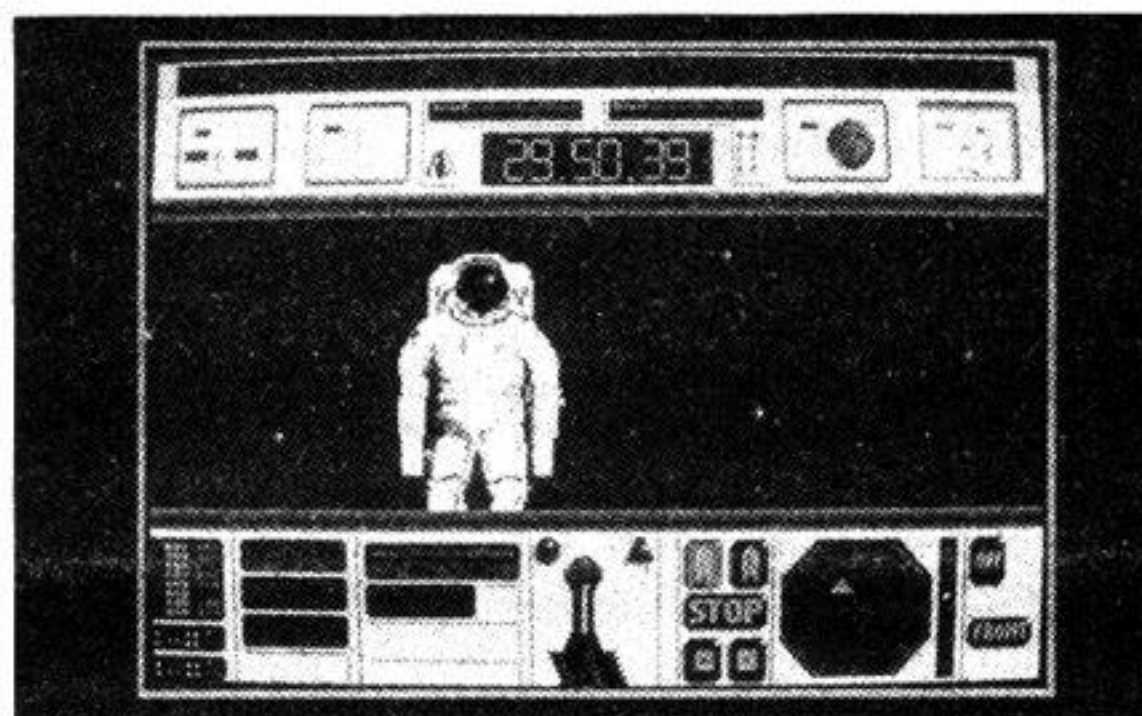
Giocando con la versione per C/64 di Arkanoid sono riuscito a realizzare un punteggio incredibile dal momento che, ad un certo punto, non "perdevo" più vite. Pur essendo contento, mi chiedo che cosa possa esser successo al programma.
Umberto Vecchi - Via Marconi 14 cap 28010 Gargallo (No)

Semplice: è probabile che la routine incaricata di intercettare le collisioni sia "saltata" senza, per questo, mandare in crash il sistema. Oppure la tua copia del gioco possiede le modifiche necessarie, appunto, a rendere infinite le vite.

Mi fa piacere, comunque, che tu abbia deciso di **vendere il tuo vecchio sistema** (C/128, drive, stampante, joy, programmi e libri vari) per passare ad Amiga. Ti consiglio, però, di impegnarti nella programmazione e di non perder tempo ad ammazzare marzianetti (ma che ti hanno fatto di male?).

Una sola testa

Possiedo un C/64 ed un drive compatibile OC-118N. Ho letto, tempo fa, che era possibile attivare la seconda testina del drive 1571 mediante alcuni comandi; questi, però, non funzionano sul mio drive. E' possibile entrare in possesso di una routine che sia in grado di svolgere il compito?
(Anonimo del tardo 900)

Il drive 1571 è dotato di due testine che vengono automaticamente rese disponibili operando **solo** con il C/128.

Tuttavia, lavorando con il C/64, è possibile avere a disposizione anche la seconda testina del drive impartendo opportuni comandi (riportati, se non erro, anche sul libretto di istruzioni). Il **1541** (ed il modello **OC-118**, che lo simula) possiedono, invece, una sola testina di lettura / scrittura e la procedura, quindi, non può funzionare nemmeno se piangi in cinese.

C/64 più veloce

Ho da poco acquistato un C/64 e ho sentito dire che esiste una particolare cartuccia (Speed Dos) capace di aumentare la velocità di lettura e scrittura. Di che si tratta?
(Anonimo 2)

Lo **Speed Dos** appartiene ad una categoria di cartucce, ormai classica, che utilizza anche la porta parallela per aumentare la velocità di trasferimento dati da e verso il drive (**non** funziona, quindi, con il registratore). Le sue caratteristiche sono davvero notevoli, ma il suo successo commerciale è risultato limitato dal momento che, per installare le schede elettroniche aggiuntive, è necessario manomettere sia il computer che il drive.

In seguito alla sua diffusione sono stati sviluppati altri sistemi che risiedono nelle Rom di cartucce aggiuntive che, come tali, possono essere inserite ed estratte dalla porta di espansione senza alcuna difficoltà.

Tali cartucce, inoltre, contengono anche comandi speciali per aumentare la velocità

di trasferimento non solo con il drive ma anche con il registratore (**Turbo Tape**). Di solito, se non bastasse, offrono anche la **possibilità di copiare programmi, pur se protetti, da nastro/disco su nastro/disco**.

Un qualsiasi rivenditore specializzato ti potrà far scegliere tra almeno una dozzina di cartucce tutto-fare diverse (si fa per dire) tra loro.

Senza intoppi

Ho trascritto il listato "Il C/64 si trasforma in orologio - sveglia" (vedi C.C.C. n. 74 ma il programma spesso non funziona correttamente. Ho quindi apportato alcune modifiche che ritengo possano interessare altri utenti.

"E' sufficiente - continua il nostro lettore - porre prima della routine preparatoria (ovvero quella che dirotta i vettori IRQ ed altri) l'istruzione Assembly Jsr $FF5B che provvede, credo, ad inizializzare il Vic II. Insomma il listato va così modificato:

```
JSR $FF5B
SEI
LDA (eccetera)
```

Il "trucco" funziona con tutti i programmi che alterano il raster interrupt.

Da Ram a Rom

Vorrei trasformare una zona Ram in Rom, dopo avervi trascritto un programma. E' possibile?
(Anonimo 3, sfida finale)

Assolutamente no, diamine! Le memorie Ram e Rom sono circuiti elettronici (hardware, cioè roba dura) e non software (roba etèrea). E' ben vero che con il C/64 è stato fatto di tutto, ma pretendere la moltiplicazione dei pani, dei pesci e dei chip potrebbe apparire presuntuoso...

Secondo me...

Egregio sig. de Simone

Scrivo a Lei in quanto direttore di C.C.C., **una** rivista che parla di prodotti Commodore, che fino a qualche mese fa consideravo **La** rivista, italiana per eccellenza, che mi ha aiutato non poco ad imparare la programmazione ed a scoprire tanti piccoli (e grandi) trucchi sui miei due computers del cuore (C/16 e, soprattutto, C/128).

Purtroppo, però, oggi ho comprato il **n. 74** e... orrore! Una blasfema fotografia con sotto scritto *"Al C/128 resta solo l'Inps"*. Ma, dico! Siamo pazzi? Non mi sono nemmeno sprecato ad usare il mio W/P, per scrivere questa lettera, in quanto C.C.C. non merita più simili attenzioni, e con questa mia presente temo altresì di aver sprecato un foglio di carta (non, se può aiutare il C/128 a sopravvivere) altrimenti utilizzabile per scrivere a qualche ragazza.

Poi, sfogliando Campus, **doppio orrore**: quella che era la mia routine di hard copy, che avevo elaborato (e vi avevo spedito) mesi e mesi fa, corredata di articolo in formato Speed Script (e ci eravamo anche messi d'accordo per telefono) e che funzionava anche su Mps-802.

Questa non vuole essere un'accusa all'autore Asruffi, in quanto non potrebbe mai essere venuto a contatto della mia routine.

E mi rendo conto che il mio articolo (forse) era scritto male; però almeno una telefonata per dirmi *"L'idea era buona, ma l'articolo è scritto da cani"* potevate anche farmela. Lo avrei certo più gradito, che non questo silenzio, in nome dell'omertà.

Dopodichè, **triplo orrore!**, *"Come diventare pirati fregando Chris Butler e vivere felici"*.

Per non parlare poi delle contraddizioni presenti in tutta la rivista (poche pagine dopo la fucilazione del C/128 lo definite una macchina *meravigliosa*). Infine, nei miei programmi (quelli della vita, non del computer) non c'è certo l'acquisto di un Amiga (il 98% dei suoi utenti lo usa solo per giocare, esattamente come il C/64. Non mi dica che non lo sapeva...) e non butterò via il C/128 finchè questo si accenderà e sarà in grado di far girare programmi (pochi ma) ottimi: il Geos, il Superscript...

Infine avete definito antistorico l'acquisto di un C/64 + drive. Da notare: prezzo di un sistema C/64 + drive, *per giocare*, L. 750.000 (max); costo di Amiga + drive esterno + espansione, sempre *per giocare*, L. 1.600.000. Che affare!!.

Personalmente ritengo antistorica la vostra riluttanza nel recensire giochi per C/64 che, in questo periodo, sta vivendo una vera e propria seconda giovinezza (Microsoccer, R-Type, Retrogade, Turrican e X-Out ne sono la prova, e la produzione continua di software per gli 8 bit in generale ne è la conferma).

Io non uso il computer per scrivere la Divina Commedia, impaginandola e corredandola di immagini, nè lo uso solo per giocare, per quel che mi serve, insomma (lo uso come w/p) il mio C/128 con drive e stampante è ancora un'ottima soluzione. Certo che non consiglierei a nessuno l'acquisto di un sistema come il mio, ma non lo ritengo certamente morto.

La vostra rivista si sta trasformando in una sorta di Commodore Gazette (censurare in caso di pubblicazione): 6000 lire per un Amiga Gazzette.

P.S.: Senza rancore, ma quella nota *"Non scriveteci siete una rivista fantastica, lo sappiamo già"* è del tutto fuori luogo.

Paolo Besser - Vigevano

La ringrazio per avermi consentito, ancora una volta, di dimostrare chiaramente che su queste pagine c'è spazio anche per chi la pensa in modo diametralmente opposto alla nostra "linea" editoriale. Quante riviste avrebbero pubblicato una lettera del genere, senza apportare la minima modifica o censura?

Perfino negli articoli dei collaboratori, eventuali "entusiasmi" per macchine obsolete (confusi come contradditori dal sig. Besser) vengono lasciati inalterati.

Mi permetto di ricordare, a proposito, che il collaboratore Luca Viola (un altro di quelli incriminati nella lettera), ex-campione del software 128, ha abbandonato il C/128 per un sistema più attuale.

Senza rancore, per carità... (A. d. S.)

SYSTEMS EDITORIALE PER TE

La voce
Aggiunge al C/64 nuovi comandi Basic che consentono sia di far parlare il computer, sia di farlo Cantare! Diversi esempi allegati.

Cassetta: L. 12000 - Disco: L. 15000

Raffaello
Un programma completo per disegnare, a colori, con il C/64: linee, cerchi, quadrati, eccetera. Valido sia per disegno a mano libera che geometrico.

Cassetta: L. 10000

Oroscopo
Devi solo digitare la data di nascita e le coordinate geografiche del luogo che ti ha dato i natali. Vengono quindi elaborate le varie informazioni (case, influenze dei segni astrali, eccetera) e visualizzato un profilo del tuo carattere. Valido per qualsiasi anno, è indicato sia agli esperti sia ai meno introdotti. E' allegata una tabella delle coordinate delle più note città italiane e l'elenco delle ore legali in Italia dal 1916 al 1978.

Cassetta: L. 12000 - Disco: L. 12000

Computer Music
Cassetta contenente numerosi brani di successo da far eseguire, in interrupt, al tuo C/64 sfruttando, fino in fondo, il suo generatore sonoro (SID).

Cassetta: L. 12000

Gestione Familiare
Il più noto ed economico programma per controllare le spese e i guadagni di una famiglia.

Cassetta: L. 10000 - Disco: L. 10000

Banca Dati
Il più noto ed economico programma per gestire dati di qualsiasi natura.

Cassetta: L. 10000 - Disco: L. 10000

Matematica finanziaria
Un programma completo per la soluzione dei più frequenti problemi del settore.

Cassetta: L. 10000 - Disco: L. 20000

Analisi di bilancio
Uno strumento efficace per determinare con precisione i calcoli necessari ad un corretto bilancio.

Cassetta: L. 10000 - Disco: L. 20000

Corso di Basic
Confezione contenente quattro cassette per imparare velocemente le caratteristiche delle istruzioni Basic del C/64 e i rudimenti di programmazione. Interattivo.

Cassetta: L. 19000

Corso di Assembler
Un corso completo su cassetta per chi ha deciso di abbandonare il Basic del C/64 per addentrarsi nello studio delle potenzialità del microprocessore 6502. Interattivo.

Cassetta: L. 10000

Logo Systems
Il linguaggio più facile ed intuitivo esistente nel campo dell'informatica; ideale per far avvicinare i bambini al calcolatore.
Diversi esempi allegati.

Cassetta: L. 6500

Compilatore Grafico Matematico
Uno straordinario programma compilatore, di uso semplicissimo, che permette di tracciare, sul C/64, grafici matematici Hi-Res ad altissima velocità. Esempi d'uso allegati.

Cassetta: L. 8000

Emulatore Ms-Dos e Gw-Basic
Un prodotto, unico nel suo genere, che permette di usare, sul C/64 dotato di drive, la sintassi tipica del più diffuso sistema operativo del mondo. Ideale per studenti.

Solo su disco: L. 20000

Emulatore Turbo Pascal 64
Permette di usare le più importanti forme sintattiche del linguaggio Turbo Pascal (anche grafiche!) usando un semplice C/64 dotato di drive.
Ideale per studenti.

Disco: L. 19000

Speciale drive
Questo speciale fascicolo costituisce una guida di riferimento per le unità a disco del C64/128.
Comprende anche un velocissimo turbo-disk più la mappa completa della memoria del drive.

Fascicolo + disco: L. 12000

Utility 1
Un dischetto pieno zeppo di programmi speciali per chi opera frequentemente con il drive.

Disco: L. 12000

Utility 2
Seconda raccolta di utility indispensabili per realizzare sofisticate procedure di programmazione.

Disco: L. 15000

Graphic Expander 128
Per usare il C/128 (in modo 128 e su 80 colonne) in modo grafico Hi-res. Aggiunge nuove, potenti istruzioni Basic per disegnare in Hi-Res con la massima velocità in modalità 80 colonne.

Disco: L. 27000

Directory
Come è noto, a partire dal N. 10 di "Software Club" (la rivista su disco per l'utente dei "piccoli" computer Commodore), vengono riportati tutti i listati, in formato C/64-C/128, pubblicati su "Commodore Computer Club".
In precedenza tali listati venivano inseriti, mensilmente, in un dischetto, di nome "Directory", che oltre ai programmi di C.C.C. ospitava decine di altri file tra cui musiche nell'interrupt, giochi, listati inviati dai lettori e altro.
Ogni disco, dal prezzo irrisorio, contiene quindi una vera miniera di software. Ordinando i dischetti di "Directory" si tenga conto che al N. 1 corrispondeva il contenuto del N. 34 di "Commodore Computer Club", al N. 2 il N. 35 e così via.

Ogni dischetto: L. 10000

Super Tot '64
La nuova e completa edizione del programma Tot 13 con tutti i sistemi di riduzione e di condizionamento.
Ampia sezione dedicata alla teoria.

fascicolo + disco: L. 15000

Amiga Totospeed
Finalmente anche per Amiga un programma orientato alla compilazione delle schedine totocalcio.
Fai tredici conil tuo Amiga.

disco: L. 20000

SYSTEMS EDITORIALE PER TE

Disk'o'teca

Grazie a questa nutrita raccolta di brani musicali potrete divertirvi ascoltando i migliori brani prodotti dai vostri beniamini, oltre a una serie di composizioni prodotte "in casa".
In omaggio un bellissimo poster di Sting.

Disco: L.15.000

Assaggio di primavera

Esclusivo!
In un'unica confezione potrete trovare ben due cassette di videogiochi assieme a un comodo e funzionale joystick.

Cassette: L. 15.000

LIBRI TASCABILI

64 programmi per il C/64

Raccolta di programmi (giochi e utilità) semplici da digitare e da usare. Ideale per i principianti. **(126 pag.)**

L. 4800

I miei amici C/16 e Plus/4

Il volumetto, di facile apprendimento, rappresenta un vero e proprio mini-corso di Basic per i due computer Commodore. Numerosi programmi, di immediata digitazione, completano la parte teorica. **(127 pag.)**

L. 7000

62 programmi per C/16, Plus/4

Raccolta di numerosi programmi, molto brevi e semplici da digitare, per conoscere più a fondo il proprio elaboratore.
Ideale per i principianti. **(127 pag.)**

L. 6500

Micro Pascal 64

Descrizione accurata della sintassi usata dal linguaggio Pascal "classico". Completa il volume un programma di emulazione del PL/O sia in formato Microsoft sia in versione C/64 (da chiedere, a parte, su disco). **(125 pag.)**

L. 7000

Dal registratore al Drive

Esame accurato delle istruzioni relative alle due più popolari periferiche del C/64. Diversi programmi applicativi ed esempi d'uso. (94 pag.)

L. 7000

Il linguaggio Pascal

Esame approfondito della sintassi usata nel famoso compilatore. (112 pag.)

L. 5000

Simulazioni e test per la didattica

Raccolta di numerosi programmi che approfondiscono e tendono a completare la trattazione già affrontata sul precedente volume. **(127 pag.)**

L. 7000

Dizionario dell'Informatica

Dizionario inglese-italiano di tutti i termini usati nell'informatica. (Edizione completa). **(385 pag.)**

L. 10000

Word processing: istruzioni per l'uso

Raccolta delle principali istruzioni dei più diffusi programmi di w/p per i sistemi Ms-Dos: Word-Star, Samna, Multimate Advantage, Word 3. **(79 pag.)**

L. 5000

Unix

Un volumetto per saperne di più sul sistema operativo professionale per eccellenza.
Un necessario compendio per l'utente sia avanzato che inesperto **(91 pag.)**

L. 5000

ABBONAMENTO

Commodore Computer Club
11 fascicoli: L. 50.000

ARRETRATI

Ciascun numero arretrato di C.C.C. L. 6.000

Come richiedere i prodotti Systems

Coloro che desiderano procurarsi i prodotti della Systems Editoriale devono inviare, oltre alla cifra risultante dalla somma dei singoli prodotti, L. 3500 per spese di imballo e spedizione, oppure L. 6000 se si desidera la spedizione per mezzo raccomandata.
Le spese di imballo e spedizione sono a carico della Systems se ciascun ordine è pari ad almeno L. 50000.
Per gli ordini, compilare un normale modulo di C/C postale indirizzato a:

C/C Postale N. 37 95 22 07
Systems Editoriale Srl
Via Mosè, 22
20090 Opera (MI)

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento"), non solo il vostro nominativo completo di recapito telefonico, ma anche i prodotti desiderati ed il tipo di spedizione da effettuare.
Per sveltire la procedura di spedizione sarebbe opportuno inviare, a parte, una lettera riassuntiva dell'ordine effettuato, allegando una fotocopia della ricevuta del versamento.
Chi volesse ricevere più celermente la confezione deve inviare la somma richiesta mediante assegno circolare, oppure normale assegno bancario (non trasferibile o barrato due volte) intestato a:

Systems Editoriale
Milano

COME INDEBITARSI PER L'ACQUISTO DI UN'AUTO

Per chi inizia, ecco l'occasione buona per usare il nuovo computer (C/64 oppure Ms/Dos compatibile): vediamo come individure l'auto dei nostri sogni. E, soprattutto, quanto pagare al mese per acquistarla...

di Egidio Stringhini

In queste pagine affronteremo due problemi, praticamente inscindibili per chi decide di effettuare una spesa di una certa rilevanza.

..Come noi rimettiamo i nostri debitucci

Supponiamo che, ad esempio, in data 14 ottobre 1990 chiediamo un prestito di lire 2.500.000 al tasso d'interesse semplice del 10% e ci impegniamo a restituire, la cifra avuta in prestito, il giorno 27 febbraio 1991. Questo appena esaminato è un classico problema di ragioneria: ci si chiede quale somma **MO** dovrò restituire, comprensiva del capitale **C** (L. 2.000000) e degli interessi semplici **IN** maturati nel periodo compreso tra le due date al tasso **TA** (10%). Non disponendo del computer, dovremo:

- Procurarci un calendario e contare i giorni **X**, esclusi il primo e l'ultimo (135, nell' esempio specifico; contare per credere).

- Calcolare l'interesse semplice maturato nel periodo usando la semplice formula...

IN = C * X * TA / 100 / 365

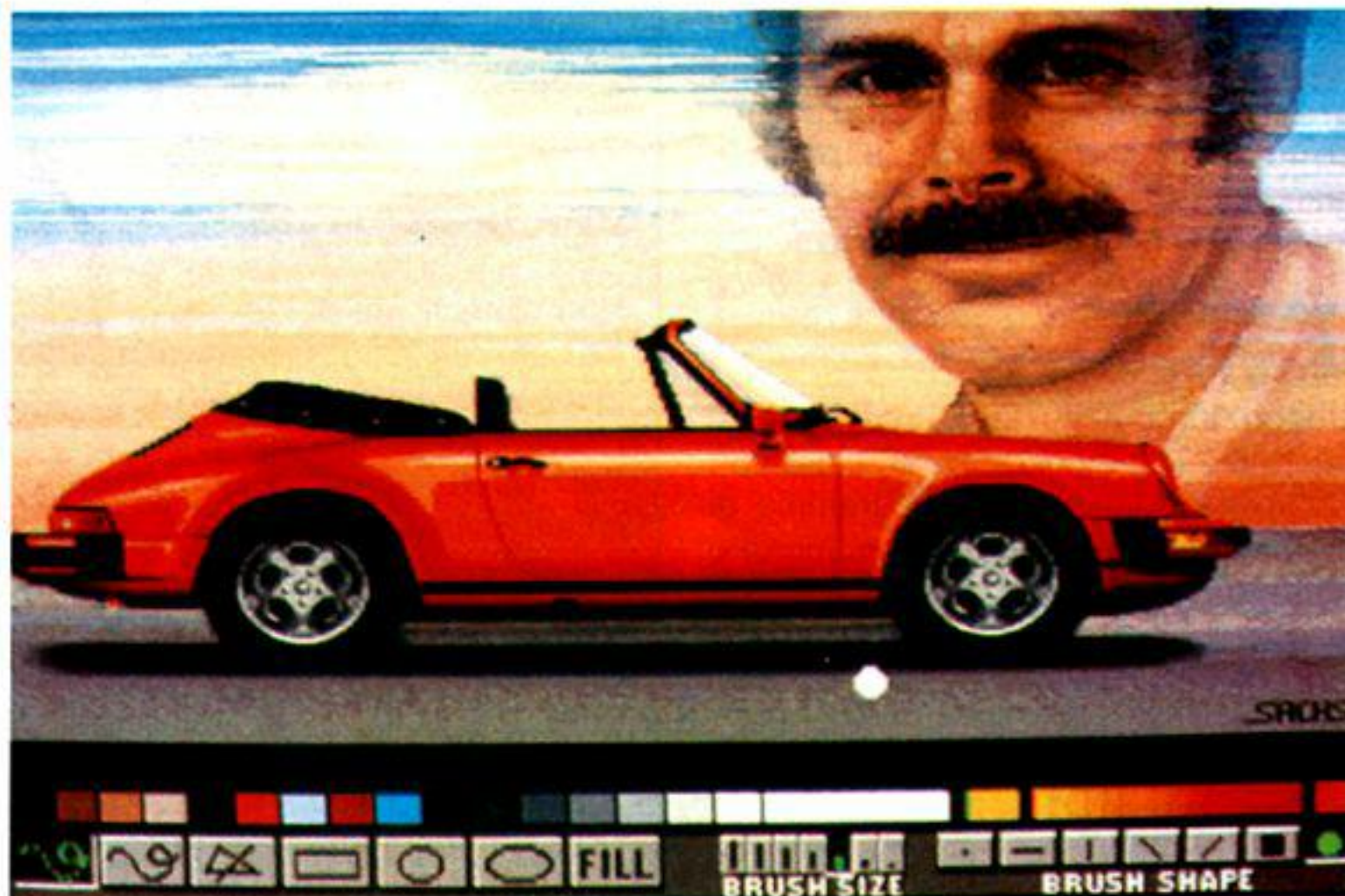
... che fornisce, sempre nel caso di prima, L. 92.465.

- Determinare quindi il *montante* **MO** come somma del capitale **C** e dell'interesse **IN** (L. 2.592.465 nell'esempio).

Vediamo ora di *implementare* (= trasportare su computer) l'intera procedura.

Nel primo programma di queste pagine è implementato anche un algoritmo per determinare il numero di giorni compresi tra due date, iniziale e finale, che potrà essere facilmente "estratto" per inserirlo in altri programmi scritti in Gw-Basic.

Righe al setaccio

Le righe 170, 190 e 210 del programma richiedono la digitazione della data d'inizio del prestito (è compreso, nei messaggi, esempi di digitazione per facilitare la battitura). Nella riga 170, e nella 190, il valore corrispondente, rispettivamente, ai giorni ed al mese, devono essere, al massimo, di due cifre;

nella riga 210, invece, l'anno **RR$** va espresso con quattro cifre. Opportuni controlli, del resto non molto sofisticati, provvedono affinchè le cifre siano digitate correttamente.

Degli anni associati alle stringhe **RR$** e **FF$** vengono considerate (vedi istruzione **Mid$** di righe 220 e 290) le ultime cifre per il fatto che le formulette (algoritmo) che calcolano il numero dei giorni, richiedono solo le ultime tre cifre (990 invece di 1990). Si potrebbe obiettare che si poteva digitare, fin dall'inizio, una cifra di tre caratteri (990); tuttavia può risultare più comodo, per l'utente, digitare l'anno per intero.

Se, alla richiesta di conferma dei dati inseriti (effettuata dalla linea 320) la risposta è negativa, la riga 340, rimandando alla riga 130, farà ripetere la procedura.

Il blocco di linee 350 / 470 costituisce l'algoritmo per la determinazione dei giorni **X** di durata del prestito. La riga 450 fornisce il numero di giorni, diminuiti del primo e dell'ultimo.

Nella 490 si attiva il calcolo dell'interesse **IN** e della somma **MO** che risulta, appunto, eguale al capitale più l'interesse maturato.

Le righe comprese tra 510 e 640 provvedono a visualizzare i risultati.

Consigli

Il principiante, cui sono dedicate queste pagine, deve fare attenzione a digitare con la massima attenzione, riga dopo riga, i listati pubblicati. Alla fine di ogni riga è indispensabile premere il tasto **Return** per confermare quanto digitato.

Nel caso in cui, dopo il **Run**, dovessero comparire segnalazioni di errore, si provveda ad eseguire la visualizzazione della linea incriminata (esempio: **List 320**) e provvedere ad apportare le dovute modifiche in modo che ciò che compare sul video sia rigorosamente identico a ciò che compare sulla rivista.

Chi possiede il **Commodore 64** deve "lanciare" l'emulatore di **Gw-Basic** prima di digitare (e salvare) i listati. In caso contrario comparirà una segnalazione di *Syntax Error* in riga 130.

Chi non disponesse dell'emulatore, deve cancellare la riga 130, o meglio digitare il carattere di doppio punto...

130 :

...subito dopo, appunto, il numero di riga 130 prima di premere il tasto Return.

Si presuppone che il lettore, benchè superprincipiante, conosca le regole più elementari per caricare, salvare e lanciare un programma scritto in Gw-Basic. In caso contrario si provveda a leggere quanto riportato sul manuale di istruzione del linguaggio interprete in proprio possesso.

Di tre macchine, quale ?

Questo secondo programma determina quale, fra tre modelli di macchina, è il più conveniente dal punto di vista economico. Esso si basa sui seguenti ragionamenti:

1 Per acquistare una macchina è necessario prelevare dalla banca la cifra necessaria che, da questo momento, non maturerà più gli interessi che prima fruttava al tasso percentuale **IN**.

Anche nel caso di ricorso ad un prestito, gli interessi graverebbero comunque, dal momento che è necessario pagarli al mutuatario (società, o banca, che finanzia il prestito).

2 Uno studio accurato richiede, inoltre, di prevedere il numero (**KM**) di chilometri che si prevede vengano percorsi in un anno.

3 Altro dato da stabilire è il numero (**AN**) di anni d'uso della macchina, prima di rivenderla per acquistarne un'altra.

A questo punto, ricorrendo a riviste specializzate, rivenditori, depliants e così via, si può stabilire, per ciascuno dei tre modelli, quanto segue:

CO = costo d'acquisto (in lire).

RE = valore residuo della macchina, ossia la cifra ottenuta rivendendo la macchina dopo gli anni d'uso precisati.

CU = consumo di carburante, sia esso benzina che gasolio, espresso in litri / 100 Km.

UC = prezzo unitario del carburante espresso in lire / litro.

AS = prezzo annuale dell' Assicurazione.

BO = prezzo annuale del bollo di circolazione (o meglio: tassa di possesso).

E' ovvio che il costo d'acquisto, "depurato" del valore residuo, sarà **CO - RE**. Per ogni anno d'uso esso graverà per una somma **AM** che chiameremo Ammortamento Annuale determinato da: AM = (CO - RE) / AN

Sofisticando

Il secondo programma pubblicato ("Macchine") affronta, in parte, una problematica decisamente più complessa. Il lettore volenteroso può ampliare la casistica, includendo le corrispondenti routines, schermate di input e di output. Ad esempio, la procedura descritta potrebbe esser considerata come base di partenza per problemi un po' particolari, come il seguente:

"Di solito utilizzo un'automobile con la quale, consumando un litro di benzina ogni 10 Km, percorro 15000 Km all'anno.

Di quest'auto non posso fare a meno e, di conseguenza, le spese fisse (assicurazione e bollo) non possono essere eliminate dal bilancio.

Allo scopo di risparmiare sul carburante decido di acquistare una motocicletta che mi limiterò ad utilizzare, per motivi... climatici, per un periodo limitato (primavera - estate) durante i quali percorrerò non più di **Y** kilometri.

Quale deve essere la cifra da spendere globalmente per la motocicletta (prezzo di acquisto, bollo, assicurazione, eventuale casco, eccetera) affinchè il risparmio annuale sul carburante sia reale?" Ai lettori più bravi, ovviamente, il compito di rispondere, magari dopo aver rispolverato le nozioni relative al *Break Even Point* (altro argomento di ragioneria...).

```
100 REM**********************************
110 REM    MACCHINE    3 Macchine *
120REM-----------------------------INPUT
130 CLS: PRINT: PRINT: PRINT: PRINT: PRINT
140 INPUT "INTERESSI % ES: 10.5 ="; IN
150 INPUT "KM ANNUALI  ES: 15000 ="; KM
160 INPUT "ANNI D'USO  ES: 5   ="; AN
170 PRINT: PRINT: PRINT: PRINT
180 INPUT "CONFERMI S/N ="; C1$
190 IF C1$ ="S" OR C1$ ="s" THEN GOTO 210
200 GOTO 130: REM---------------------INPUT
210 INPUT "QUANTE AUTO (1 / 3) "; QA
220 IF QA <1 OR QA > 3 THEN 210
230 FOR I = 1 TO QA
240 CLS: PRINT: PRINT: PRINT: PRINT
250 PRINT "MACCHINA"; I: PRINT "---------- "
260 INPUT "MODELLO MAX 9 CARATT ="; W$ (I)
270 IF LEN (W$ (I) ) > 9 THEN PRINT "ACCORCIA "
280 IF LEN (W$ (I) ) > 9 THEN GOTO 260
290 INPUT "COSTO D'ACQUISTO - (L.) ="; CO (I)
300 INPUT "VALORE RESIDUO - (L.) ="; RE (I)
310 INPUT "CARBURANTE LT /100 KM ="; CU (I)
320 INPUT "CARBURANTE LIRE / LT  ="; UC (I)
330 INPUT "ASSICURAZ. LIRE/ ANNO ="; AS (I)
340 INPUT "BOLLO   LIRE/ ANNO ="; BO (I)
350 PRINT: PRINT: INPUT "CONFERMI S/N ="; C2$
360 IF C2$ ="S" OR C1$ ="s" THEN GOTO 380
370 GOTO 240
380 NEXT I: REM----------------CALCOLETTI
390 FOR I = 1 TO 3
400 AM (I) = (CO (I) - RE (I) ) /AN
410 NI (I) = AM (I) * IN/100
420 CC (I) = CU (I) * UC (I) * KM * 1.15 / 100
430 T (I) = AM (I) + NI (I) + CC (I) + AS (I) + BO (I)
440 NEXT I: CLS: PRINT: PRINT: PRINT
450 REM----------SCHERMATA DEI RISULTATI
460 FOR I = 1 TO 3: PRINT TAB (2+10*I);
470 PRINT W$ (I);: NEXT I: PRINT
480 PRINT TAB (12) "--------- ---------";
490 PRINT TAB (32) "--------- "
500 PRINT "AMMORT.TO";
510 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
520 PRINT INT (AM (I) );: NEXT I: PRINT
530 PRINT "INTERESSI";
540 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
550 PRINT INT (NI (I) );: NEXT I: PRINT
560 PRINT "CARBUR.TE";
570 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
580 PRINT INT (CC (I) );: NEXT I: PRINT
590 PRINT "ASSICURAZ";
600 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
610 PRINT INT (AS (I) );: NEXT I: PRINT
620 PRINT "BOLLO....";
630 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
640 PRINT INT (BO (I) );: NEXT I: PRINT
650 PRINT "TOT/ANNO.";
660 FOR I = 1 TO 3: PRINT TAB (1 + 10 * I);
670 PRINT INT (T (I) );: NEXT I: PRINT
680 '---------------RIORDINO CLASSIFICA
690 PRINT: FOR K = 1 TO 3: FOR I = 1 TO 3
700 IF T (I) < T (I-1) THEN GOSUB 770
710 NEXT I: NEXT K: REM----------SCHERMATA
720 PRINT "SPESE ANNUALI (LIRE) "
730 FOR I = 1 TO QA
740 PRINT TAB (4); W$ (I); TAB (19);
750 PRINT INT (.5+T (I) ): NEXT I: END
760'---------------------------SCAMBIO
770 AA = T (I-1): T (I-1) = T (I): T (I) = AA
780 BB$ = W$ (I-1): W$ (I-1) = W$ (I): W$ (I) = BB$
790 RETURN: REM**************************
800 END
```

```
100 REM ******************
110 REM * INTERESSI Calcolo Interessi *
120 REM * bancari tra 2 date *
130 CLS: PRINT: REM - - - - - - - - - - - - -
140 PRINT "CALCOLO INTERESSI BANCARI "
150 PRINT " MATURATI TRA DUE DATE": PRINT
160 PRINT: PRINT "DATA INIZIALE": REM - - - -
170 INPUT " GIORNO Es: 9 ="; T
180 IF T < 1 OR T > 31 THEN GOTO 170
190 INPUT " MESE Es: 5 ="; S
200 IF S < 1 OR S > 12 THEN GOTO 190
210 INPUT " ANNO Es: 1990 ="; RR$
220 R = VAL (MID$ (RR$, 2) ) : PRINT
230 PRINT "DATA FINALE": REM - - - - - - - -
240 INPUT " GIORNO Es: 9 ="; O
250 IF O < 1 OR O > 31 THEN GOTO 240
260 INPUT " MESE Es: 11 ="; V
270 IF V < 1 OR V > 12 THEN GOTO 260
280 INPUT " ANNO Es: 1990 ="; FF$
290 F = VAL (MID$ (FF$, 2) ) : PRINT
300 INPUT " INTERESSE % ES: 7.5 ="; TA
310 INPUT " CAPITALE (LIRE) .... ="; C
320 PRINT: INPUT " CONFERMI S / N ="; C1$
330 IF C1$ =" S " OR C1$ =" s " THEN GOTO 350
340 GOTO 130: REM - - - - - - CALCOLO GIORNI X
350 H = R: G = S: I = T
360 IF G - 3 > = 0 THEN GOTO 460
370 G = G + 13: H = H - 1
380 I = INT (365.25 * H) + INT (30.6 * G) + I
390 I = I - INT (H / 100) + INT (H / 400) - 306 - 122
400 J = I: H = F: G = V: I = O
410 IF (G - 3) > = 0 THEN 470
420 G = G + 13: H = H - 1
430 I = INT (365.25 * H) + INT (30.6 * G) + I
440 I = I - INT (H / 100) + INT (H / 400) - 306 - 122
450 X = I - J - 1: GOTO 480
460 G = G + 1: GOTO 380
470 G = G + 1: GOTO 430
480 REM - - - CALCOLO INTERESSI E MONTANTE
490 IN = C * TA * X / 100 / 365: MO = C + IN
500 CLS: PRINT: REM - - - - VIDEATA RISULTATI
510 PRINT: PRINT "DATA INIZIALE "
520 PRINT "GIORNO....... ="; T
530 PRINT "MESE.......... ="; S
540 PRINT "ANNO.......... ="; RR$
550 PRINT: PRINT "DATA FINALE "
560 PRINT "GIORNO....... ="; O
570 PRINT "MESE.......... ="; V
580 PRINT "ANNO.......... = "; FF$
590 PRINT: PRINT "GIORNI - ESCLUSI "
600 PRINT "PRIMO E ULTIMO ="; X
610 PRINT: PRINT "INTERESSE % ="; TA; " % "
620 PRINT "CAPIT. (L.) ="; C
630 PRINT "INTER. (L.) ="; INT (IN), "(" IN ")"
640 PRINT "MONTA. (L.) ="; INT (MO), "(" MO ")"
650 END
```

A tale onere, annuale, andranno aggiunti gli interessi passivi...

NI = AM * IN / 100

La determinazione della spesa annuale per il carburante si effettua con la formuletta...

CC = CU * UC * KM * 1.15 / 100

...nella quale il fattore **1.15** ha la funzione di maggiorare i consumi teorici, forniti dalle case costruttrici, del 15% in modo da adeguarli ad una realtà più credibile. Per modificare tale fattore si può agire, comunque, sulla riga 420 del programma alterando la cifra **1.15** ivi presente.

L'ultimo calcolo riguarda l'onere annuale totale determinato come somma di Ammortamento + Interessi + Carburante + Assicurazione + Bollo, cioè:

T = AM + NI + CC + AS + BO

La vettura che presenta minore incidenza totale annuale sarà, ovviamente, quella più conveniente dal punto di vista economico.

Righe al setaccio

Le righe 140, 150, 160 servono per l'inserimento del tasso di interesse percentuale (**IN**), del percorso annuale previsto **KM** e del numero di anni d'uso **AN**.

Le linee 190 e 200 permettono l'eventuale correzione dei dati inseriti in quanto rimandano all'inizio della procedura.

Esaminiamo ora il *Loop* del blocco 210 / 380. Esso richiede i dati riguardanti ciascuno dei tre modelli di macchina, indicizzando tali dati sulla variabile **I**. I dati richiesti sono:

W$ (I): modello della macchina: riga 260; CO (I): prezzo d'acquisto in lire. RE(I): valore residuo; CU(I): consumo unitario carburante (litri / 100 KM); UC(I): prezzo unitario carburante (lire / litro); AS(I): prezzo annuale assicurazione; BO(I): prezzo annuale.

Nel caso in cui il nome del modello digitato risulti più lungo di 9 caratteri, verrà rifiutato. Questa protezione serve per garantire l'allineamento verticale della schermata dei risultati, dal momento che (per motivi di compatibilità con il C/64) questa è costretta nel limite di 40 caratteri orizzontali.

Se, alla richiesta di conferma dei dati inseriti (linea 360), la risposta fosse negativa, la riga 370 rimanderebbe il computer all'inizio per ripetere le operazioni relative al modello di auto interessato, in quel momento, dall'elaborazione.

Segue l'esecuzione dei calcoli (indicizzati con la variabile **I**) relativi a ciascun modello di macchina:

AM (I): ammortamento annuale; NI(I): interessi passivi annuali; CC(I): prezzo carburante annuo; T(I): onere totale annuale.

Con il blocco di righe 460 / 670 si visualizza la schermata dei risultati nella quale vengono riportate le incidenze di costo di gestione relative a ciascuno dei tre modelli.

In fondo alla schermata appare la classifica degli oneri, in ordine decrescente, delle tre auto (righe 720 / 750).

Questa risulta impostata dal Loop 690 / 710 con la routine di scambio di cui alle linee 770 / 790.

TASTO CHE VA, VIDEO CHE VIENE

Una comoda routine per gli estimatori delle 80 colonne, e una manciata di locazioni anche per chi non le usa

di Domenico Pavone

L'utente medio del C/128, con le solite eccezioni a conferma della regola, prima o poi riorganizza il proprio sistema hardware in modo da implementare la visualizzazione dello schermo su 80 colonne.

Senza nulla togliere al comune video da 40, i vantaggi che ne derivano sono molteplici, e, come noto, non legati esclusivamente ad un fatto visivo.

Non va dimenticato, infatti, che in modo 80 colonne è possibile sfruttare appieno la capacità del computer di operare a 2MHz (*Fast Mode*), godendo di tutte le prerogative legate al mantenimento dello schermo visibile.

Operazioni banali, ma più che frequenti, come listare un programma o spulciare una directory, assumono tutto un altro andazzo.

Il fatidico *Blank*, diventa presto uno sbiadito ricordo.

Tuttavia, per i motivi più disparati, capita spesso di doversi "spostare" da uno schermo all'altro, soprattutto quando le proprie elaborazioni tendono a sfruttare sia le 40 che le 80 colonne.

Il che si traduce, in pratica, in continue pressioni di tasti e pulsanti: prima *Escape*, poi *X*, ed infine l'interruttore sul monitor.

E non è ancora finita.

A seconda del senso dello "switch", va anche opportunamente impartito un comando *Fast* oppure *Slow*... sempre che la proverbiale pigrizia digitatoria del centoventottista non abbia il sopravvento.

Se, come sempre accade, non si è poi premuto il tastino 40 / 80 Display, al primo *Run / Stop+Restore* ci si ritrova al punto di partenza.

In definitiva, non è proprio il massimo della comodità, o, per dirlo forbitamente, la procedura non è *ergonomica*.

Per fortuna a tutto c'è rimedio, in special modo quando si ha a che fare con i nostri beneamati computer.

Se non ci hanno pensato i progettisti della macchina, siamo forse noi da meno per non provvedere da soli?

Beh, forse è opportuno non montarsi la testa, ma qualcosa possiamo davvero farla, con risvolti didattici non indifferenti, ed accontentando anche i soliti "affamati" di *Peek* e *Poke* che non dispongono di un monitor ad 80 colonne.

(QUASI) SENZA MANI

La logica premessa a quanto vedremo, può essere riassunta da un semplice interrogativo.

Visto che la tastiera del 128 è fornita di un tasto / deviatore 40 / 80 Display, praticamente utile solo in fase di avvio del sistema, perchè non affidare a lui tutto l'onere dello switch tra le due modalità, Fast / Slow compresi, senza stare a smanettare sulla tastiera?

Detto fatto.

Tuto ciò che occorre, sono una cinquantina di byte in linguaggio macchina (davvero pochi, in fondo!).

Vediamoli subito in azione, prima di esaminare una tecnica che si discosta da quella usuale per gestire la tastiera.

Si copi, dunque, il listato basic di queste pagine, e lo si salvi su disco o nastro.

In alternativa, chi ha già una certa dimestichezza con il *Monitor LM* in dotazione alla macchina, può copiare (istruzione *A 1300*...) direttamente le istruzioni *Assembly* del disassemblato (ovviamente senza i commenti), e salvare il codice macchina con...

```
100 rem----------------------------
110 rem autoswitch 40-80 colonne
120 rem con autofastmode
130 rem----------------------------
140 rem   per commodore c/128
150 rem----------------------------
160 :
170 for x=0 to 53: read a
175 poke 4864 + x, a: b = b + a
180 next: if b <> 5697 then 230
190 print chr$ (147) "switch immediato";
200 print "su tasto 40/80 colonne"
210 print "con modo fast selettivo"
220 sys 4864: end
230 print"errore nelle linee data!"
240 end
250 data 120, 169, 013, 162, 019, 141, 020
260 data 003, 142, 021, 003, 088, 096, 173
270 data 005, 213, 009, 128, 141, 005, 213
280 data 165, 215, 077, 005, 213, 048, 009
290 data 032, 159, 205, 032, 042, 192, 032
300 data 111, 205, 165, 215, 010, 144, 006
310 data 032, 179, 119, 076, 101, 250, 032
320 data 196, 119, 076, 101, 250
330 end
```

```
DISASSEMBLATO AUTOSWITCH
40-80 COLONNE

1300 SEI        Disabilita interrupt.
1301 LDA #$0D   Inserisce indirizzo
1303 LDX #$13   $130D (inizio nostra
1305 STA $0314  routine) nel vettore
1308 STX $0315  di interrupt.
130B CLI        Riabilita interruzioni.
130C RTS        Torna al basic.
--------------------------------------
130D LDA $D505  Abilita alla lettura
1310 ORA #$80   dello switch 40/80 col.
1312 STA $D505  tramite bit 7 di 54533.
--------------------------------------
1315 LDA $D7    Confronta Flag schermo
1317 EOR $D505  attivo con tasto 40/80.
131A BMI $1325  Se diseguali (=OK) esce.
--------------------------------------
131C JSR $CD9F  Elimina il cursore.
131F JSR $C02A  Cambia schermo attivo.
1322 JSR $CD6F  Riabilita il cursore.
--------------------------------------
1325 LDA $D7    Controlla bit 7 della
1327 ASL        locazione 215.
1328 BCC $1330  Se = 0 (40 col.), salta.
--------------------------------------
132A JSR $77B3  Abilita modo Fast.
132D JMP $FA65  Esce ad IRQ di sistema.
--------------------------------------
1330 JSR $77C4  Abilita modo Slow.
1333 JMP $FA65  Esce ad IRQ di sistema.
```

S "nomefine", 8, 1300, 1336

...se avete il drive; se possedete il registratore, invece:

S "nomefine", 1, 1300, 1336

Per adoperare il file così prodotto, non si dimentichi, lo stesso va caricato con Bload (per il disco) oppure...

Load "nomefile", 1, 1

...per il tape, e mandato in esecuzione con:

Sys 4864.

Con il caricatore basic, invece, tutto ciò che occorre fare è impartire *Run* (la *Sys* è già inserita nel listato).

Da questo momento in poi, basterà premere il tasto 40 / 80 Display (ed il deviatore del monitor) per cambiare schermo, ed avere (ma solo sulle 80 colonne) il modo *Fast* sempre attivo.

Si badi che il sistema si adegua immediatamente allo stato del tasto (pressato / sollevato): se, al momento della Sys (o del Run al listato) ci si trova p. es. in modo 40 colonne, ma il tastino è posizionato sulle 80 (abbassato), è quest'ultimo modo che diventerà attivo, "blankando" lo schermo disattivo (quello 40 colonne).

Nel passaggio inverso, il Fast viene disattivato, e, lasciando la routine così com'è, senza alcuna possibilità di utilizzarlo, neanche impartendo il relativo comando da tastiera (per il Fast, in fondo, c'è lo schermo a 80 colonne).

Se, comunque, si desidera disporre dei 2 MHz anche sulle 40 colonne, basta digitare da basic, possibilmente non mentre la routine è in funzione...

Poke 4891, 23

Per rendere definitiva tale modifica, va cambiata l'istruzione Assembly *131A* (presente nel disassemblato) in modo che diventi *BMI $1333*, e poi salvato il codice macchina come in precedenza descritto.

Optando per tale soluzione, dopo *Sys 4864*, e solo la prima volta che si adopera il tasto 40 / 80, il blank dello schermo 40 colonne può non essere attivato immediatamente (effetto comunque trascurabile).

La pressione di *Run / Stop+Restore* (o un *Reset*) disabilita la routine, che può comunque essere riattivata con una nuova *Sys 4864*, magari assegnata ad uno dei tasti funzione (comando *Key*).

UGUALI E CONTRARI

Alla base del funzionamento della routine, sta una adeguata manipolazione dell'*Interrupt* di sistema.

Argomento, questo, trattato ormai decine di volte nell'ambito della rivista, e che quindi non staremo qui a ribadire.

Giusto come promemoria, si può comunque dare un'occhiata alla tabella degli indirizzi utili, pubblicata a parte, che riporta anche l'*Entry Point* dell'*IRQ* del C/128 (*$FA65*) e una sua breve descrizione.

Come si può intuire, il primo compito che il nostro programma deve svolgere ad ogni ciclo di interrupt, è quello di testare la posizione del tasto 40 / 80 Display.

Per far ciò, però, non è possibile ricorrere alle più volte descritte tecniche di controllo sulla scansione della tastiera (vedi anche C.C.C. n. 67).

Questa routine di sistema, attivata ogni sessantesimo di secondo, non effettua infatti alcun controllo sul tasto che ci interessa, il cui stato (alzato / abbassato) è direttamente collegato al bit 7 del registro *$D505* (vedi tabella).

In particolare, alla condizione *giù* (80 colonne) corrisponde lo zero, mentre 1 indica che il tasto è sollevato (40 col.), senza dimenticare che la posizione fisica dello stesso non identifica necessariamente lo schermo attivo (ci si può trovare in 80 colonne, ma con il tasto sollevato).

L'informazione sull'attuale modalità video, indipendentemente dallo stato del tasto 40 / 80, è invece prelevabile da un altro bit 7, quello della locazione *$D7* (descritta in tabella).

Solo che, a confondere un po' le cose, la relazione tra condizione del bit e schermo attivo è invertita rispetto a quanto prima visto con $D505: settato indica le 80 colonne, azzerato le 40.

IL DISASSEMBLATO

Su queste premesse, seguiamo ora il disassemblato commentato.

Le prime istruzioni (1300 - 130C) dirottano il normale corso dell'Interrupt di sistema verso la nostra routine, che inizia realmente a partire da $130D.

Qui, come manovra preliminare, viene effettuato un *OR* del registro $D505, in modo da settarne il bit 7 (= porlo ad 1).

Perchè mai?

Ebbene, il comportamento del registro è diverso a seconda che si acceda ad esso in scrittura oppure in lettura.

Nel secondo caso, come già detto, è possibile rilevare lo stato del tasto 40 / 80.

Perchè l'informazione sia corretta, è però opportuno obbligare il sistema a leggere tale condizione.

Questa "forzatura", si ottiene appunto settando (in scrittura, con *STA*) il nostro caro bit 7, cosa che normalmente il

INDIRIZZI UTILI

$D7 (decimale 215)
Il bit 7 di questa locazione di pagina 0 riporta quale modalità di schermo è attiva. Azzerato, indica le 40 colonne, settato (= 1) le 80 colonne.
Da usarsi solo per conoscere il tipo di schermo attivo, mai per passare da uno all'altro: una *Poke 215, 0* (mentre ci si trova in 80 colonne), pur essendo formalmente corretta, e pur provocando un'effettivo "switch" alle 40 colonne, porterà in breve a conseguenze nefaste.
Lo stesso dicasi per *Poke 215, 128* mentre ci si trova in modalità 40 colonne.
Allo switch, infatti, deve corrispondere anche un riassetto delle variabili dello Screen Editor, del link tra le linee, ed altre cosucce amene.

$D505 (decimale 54533)
E' uno dei cosiddetti "registri" della *MMU* (Memory Management Unit).
Si occupa, tra le altre cose, della selezione tra Rom del 128 e Rom del 64, del CP/M, e della comunicazione Fast con la porta seriale.
Il suo contenuto va quindi trattato con molta cautela, ed eventuali modifiche devono essere rivolte esclusivamente al bit che interessa: nel caso della routine di queste pagine, il bit 7, connesso allo stato del tasto 40/80 colonne.
In lettura, questo bit risulterà azzerato se il tasto è abbassato (80 col.), settato (= 1) se il tasto è sollevato (40 col.).

$CD9F (decimale 52639)
L'area Rom (di bank 15) da $C000 a $CFFF, all'interno dell'interprete basic, è riservata nel C/128 allo Screen Editor.
La routine di sistema che inizia a $CD9F, elimina il cursore dallo schermo attivo.
In particolare, invocata prima dello switch tra 40 ed 80 colonne, evita che possa rimanere stampato un quadrato in reverse, "eredità" del lampeggio del cursore, nel video 40 colonne, nonchè il persistere dello stesso nella visualizzazione 80 colonne.

$CD6F (decimale 52591)
Indirizzo della Rom - routine che svolge un lavoro opposto a quella precedente.
Riabilita il cursore, facendolo riapparire nella sua precedente posizione.

$C02A (decimale 49194)
Ingresso effettivo della routine a nome *Swapper*, che effettua il passaggio da 40 ad 80 colonne e viceversa, organizzando opportunamente variabili di schermo, link, eccetera, e settando la locazione $D7.
Tale indirizzo può essere raggiunto anche tramite la tavola di vettori del Kernal, richiamando con Jsr (da LM) o con Sys (da Basic) l'indirizzo $FF5F (decimale 65375).
Un altro ingresso possibile è rappresentato da $CD2E, cui accede il sistema quando si premono i tasti Escape ed X, per giungere comunque ad un salto al nostro $C02A.

$77B3 (decimale 30643)
Inizio routine di esecuzione del comando basic Fast.

$77C4 (decimale 30660)
Inizio routine di esecuzione del comando basic Slow.

$FA65 (decimale 64101)
Indirizzo della routine di Interrupt, contenuto (in formato basso/alto) nei vettori $314 e $315.
Alterando questi ultimi, si può far "passare" il sistema, ad ogni ciclo di interruzione (60 volte al secondo), attraverso una nostra routine.
Rispettando, però, due condizioni tassative: che sia visibile in banco 15, e che la stessa si concluda con un salto (Jmp) ad $FA65.

C/128 esegue ad ogni Run / Stop+Restore. Nella routine, quindi, ad ogni ciclo di Interrupt, verrà aggiornato il bit del registro, che noi potremo così raffrontare con quello della locazione $D7.

In pratica: se i due bit sono diseguali, allora schermo attivo e condizione del tasto coincideranno.

In caso contrario, vorrà dire che vi è stato un cambiamento, ovvero che si è agito sul tasto 40/80 Display.

Le istruzioni da 1315 a 131A, si occupano di verificare la corrispondenza, o meno, tra i bit delle due locazioni prima descritte, procedendo ad un *EOR* (Or Esclusivo) tra i loro contenuti.

L'eventuale discrepanza tra i due bit 7, viene segnalata nel registro di stato con un settaggio del *flag di segno* (anch'esso il bit 7, tanto per cambiare), per cui si potrà dirottare opportunamente il flusso del programma con l'istruzione *BMI* (Salta se negativo, ovvero se il flag di segno è ad 1).

Complicazioni a parte, se si è premuto il tasto, vengono invocate le routine di sistema (descritte in tabella) per disattivare il cursore, modificare lo schermo attivo, e riabilitare il cursore (131C-1322).

Dopodichè, a seconda della condizione del bit 7 di $D7 (istruzioni 1325-1328), viene attivato il modo Fast o quello Slow (si vedano i commenti del disassemblato).

Per testare la condizione del bit, si effettua una rotazione a sinistra del valore contenuto in $D7 (*ASL* = Aritmetic Shift Left), che immagazzinerà nel flag di riporto (*Carry*) del registro di stato, il bit più a sinistra, ovvero il settimo.

In tal modo, basterà poi un *BCC* (Branch on Carry Clear = Salta se il Carry è azzerato) per decidere quale subroutine richiamare (Fast o Slow) prima della definitiva uscita al normale Interrupt ($FA65).

A seguito di tali manipolazioni, come intuitivo, la pressione di *Escape+X* diverrà insignificante.

Sarà sempre e solo il tasto 40/80 Display a decidere quale modalità di visualizzazione sarà attivata in un determinato momento.

Il programma in sé (da 130D in poi) è completamente rilocabile, ovvero può essere spostato in qualunque altra area di memoria (purchè di banco 15), ma avendo l'accortezza di usare il suo nuovo indirizzo per settare il vettore di Interrupt (istruzioni 1301-1308).

A dispetto della brevità della routine, non si può proprio dire che sia di facile comprensione, soprattutto se si è alle prime armi con l'Assembly.

D'altra parte, per rispondere al gran numero di lettori che richiedono insistentemente di trattare il linguaggio macchina anche per il C/128, l'organizzazione di questo computer è abbastanza complessa, da richiedere spesso una conoscenza molto approfondita del linguaggio.

Per consolarsi, si rivolga il pensiero a chi le stesse pene le soffre nei confronti di Amiga.

Perchè in quel caso, in tema di Assembly, applicazioni come queste sembrano proprio uno scherzo!

CAMPUS

64 / 128

18 IO DIGITO, TU COMPRIMI, LUI DECOMPRIME

Un solo articolo occupa, questo mese, l'intero inserto di Campus dedicato ai "piccoli" della Commodore; che, poi, tanto piccoli non sono, dal momento che è possibile sviluppare procedure attuate su computer più evoluti.
La novità di rilievo consiste, però, nel fatto che i listati girano perfettamente sia sul C/64 che sugli "altri" (C/128 modo 40 e 80 colonne, C/16 e perfino Plus/4!). La compressione di un testo è uno dei problemi più affascinanti che è possibile riscontrare in informatica, sia perchè costringe all'uso di sofisticati algoritmi di elaborazione (preso in esame è l'algoritmo di Houfman), sia perchè chiama in causa, indirettamente, il problema della crittografia, sia perchè, soprattutto, consente di conseguire consistenti risparmi nel caso in cui si decida di trasferire dati via modem.
La decisione di affrontare un tema così impegnativo può apparire, in effetti, in contrasto con lo spirito gaio e disimpegnato che tutte le riviste del settore seguono nel periodo estivo (e di vacanza in particolare). Ma le recenti proteste di alcuni lettori, che ci accusano (a torto) di eludere argomenti complessi, ci ha indotto a proporre 15 pagine che, volendo, possono essere lette (e... studiate) anche sotto l'ombrellone, avendo a disposizione un foglio di carta (meglio due..) ed una penna; e questo per soddisfare la sana curiosità di vedere che succede tracciando alberi e codici di vario tipo.
Nè più nè meno che se si trattasse di risolvere un puzzle della Settimana Enigmistica.

Nessun albero viene abbattuto per gli inserti di *Commodore Computer Club*, stampati su carta riciclata al 100%

IO DIGITO, TU COMPRIMI, LUI DECOMPRIME

Una procedura, complessa ma completa, per comprimere e decomprimere files di qualsiasi tipo servendosi di un semplice C/64, C/16, Plus/4 oppure C/128! Utile per diminuire le bollette SIP; e per esercitare la mente

di Claudio Baiocchi

L'uso di un modem si rivela costoso quando la quantità dei dati da inviare è notevole

Chi, disponendo di un modem, usa il telefono per scambi di programmi, è senz'altro interessato alla possibilità di abbassare gli importi delle bollette SIP; in questo articolo ci occupiamo della possibilità di **comprimere** un messaggio in modo da ridurne, spesso drasticamente, la lunghezza. L'argomento offrirà l'occasione per parlare della gestione (da Basic) di una delle più importanti strutture informatiche: quella di "**albero**".

Un ripasso sui binari

Naturalmente non sono quelli del treno! Una stringa **B$** costituita da **8** simboli scelti solo tra **0** e **1** può essere interpretata come la *rappresentazione binaria* di un numero **A** compreso tra **0** e **255**; parleremo spesso del "byte" A; o equivalentemente, del "carattere" associato, dato da chr$ (A).

Data la stringa **B$**, il valore di A si ottiene accumulando gli addendi:

128, da accettare solo se la prima cifra di B$ è un 1;

64, solo se la seconda cifra di B$ è un 1;

e così via, con gli addendi 32, 16, 8, 4, 2, 1. Si faccia bene attenzione sia al fatto che in B$ ci possono essere solo i simboli 0 e 1, sia al fatto che ci devono essere 8 simboli (cioè: non è permesso sopprimere eventuali zeri iniziali).

Viceversa, dato un byte A, la sua rappresentazione binaria può essere trovata in più modi; ad esempio ripetendo 8 volte le operazioni...

B = A + A; A = B and 255

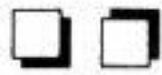
...scrivendo 1 se B > 255, 0 in caso contrario.

Supponiamo ora di voler trasmettere un lungo messaggio binario, diciamo una successione di **mille** cifre scelte tra **0** e **1**.

La strategia ora vista permette di ridurne drasticamente la lunghezza: raggruppando i bit a gruppi di 8, e sostituendo ogni gruppo con un unico carattere, l'intero messaggio originale sarà compattato in soli 125 byte. Il lettore che conosce (anche superficialmente) l'alfabeto Morse starà forse pensando che, sostituendo il "punto" con uno 0 e la "linea" con un 1, si può adattare questa tecnica per comprimere qualunque messaggio; in realtà le cose non stanno così: l'alfabeto Morse non usa *due* simboli, ma *tre:* oltre a punto e linea, c'è il *silenzio* che avvisa il ricevente che una lettera è terminata e ne sta iniziando un'altra! La compressione può invece funzionare (sia pure con qualche limitazione) se vogliamo trasmettere un **itinerario**; il messaggio...

ssdsddss

...(che significa: "al primo bivio prendi la strada di sinistra, al secondo anche, al terzo prendi quella di destra..." e così via) può essere concentrato nell'unico simbolo **chr$ (44)** avendo sostituito **s** con **0** e **d** con **1**. La limitazione cui accennavamo dovrebbe essere ovvia: se in qualche punto nel percorso c'è un incrocio (un

"trivio", anzichè un bivio) casca tutto perchè, con due soli simboli, manca la possibilità di dire: "vai dritto".

Conoscere gli alberi

Diamo ora un'occhiata alla figura 1; in essa si distinguono:

- una **radice**, indicata da una freccia;
- dei punti di biforcazione, detti **nodi**, dai quali partono due **rami** (e sempre due soli; in queste condizioni l'albero si dice binario);
- dei **nodi terminali**, ognuno contenente una lettera.

La terminologia in uso prende a prestito termini anche dall'araldica, oltre che dalla botanica: un albero genealogico (che di solito si disegna alla rovescia: la radice corrisponde al "capostipite" ed è il punto più in alto nel disegno) descrive le relazioni di parentela padre - figlio; la nostra ipotesi di avere a che fare con un albero binario dice che ogni padre ha esattamente **due** figli; sono inoltre proibiti i matrimoni tra consanguinei, cosicchè ogni figlio ha esattamente **un** genitore nell'albero (salvo il capostipite, che non ne ha). Esplicitiamo infine un'ultima proprietà, valida nell'albero di figura 1 ma falsa negli alberi genealogici: tutta l'informazione è contenuta nei nodi terminali (cioè: nei punti di biforcazione non abbiamo inserito lettere).

Una volta costruito il vettore **Codice$** come indicato in figura 1, proviamo a codificare il titolo di una canzone molto in voga la scorsa estate: "viva la mamma". Basterà eseguire:

```
Messaggio$ = "viva la mamma"
B$ = "": For X = 1 to len (Messaggio$)
A = Asc (Mid$ (Messaggio$, X) )
B$ = B$ + Codice$ (A): Next
```

E' facile verificare che il messaggio binario B$ ha lunghezza 32; compattando i bit a gruppi di 8, si perviene ai soli 4 byte 28, 46, 38, 105. La cosa sembra interessante: un messaggio di 13 simboli si è ridotto a solo 4! Il "miracolo" è naturalmente dovuto ad una oculata scelta dell'albero: si osservi, infatti, che le lettere più frequenti nel nostro messaggio (la **A**, che compare 4 volte, e la **M** che compare 3 volte) hanno un **Codice$** lungo solo due bit; mentre le altre, che compaiono più di rado, hanno un Codice$ lungo **3**.

Per ottenere buoni risultati diventa perciò essenziale scegliere bene l'albero: ogni messaggio possiede un suo "albero ottimale" che, in onore all'ideatore di una comoda stategia per ottenerlo, viene detto albero di **Huffman** del messaggio in questione. Tale strategia, illustrata nel riquadro "Dalla mamma ai numeri", nel caso di *viva la mamma*, sarà discussa e implementata più tardi; per quanto riguarda il mini programma ora visto osserviamo tuttavia che, in pratica, dovremo complicare un po' le cose: quando, abbandonato l'esempio "viva la mamma", cerchiamo di codificare un file vero, ci scontriamo con varie piccole difficoltà. In generale **Messaggio$** lo dovremo leggere, byte dopo byte, da un file su disco; non enterà tutto in una stringa; ancora peggio, la stringa B$ si gonfia molto rapidamente, perciò ogni tanto dovremo "scorciarla" traducendone le prime 8 cifre in un byte; byte che, per motivi di capienza della memoria, saremo in generale costretti ad inviare ad un file sul disco.... Non si tratta di problemi complicati, tuttavia, vista la proverbiale lentezza del Basic Commodore nel colloquio col disco, l'unica soluzione consiste nello scri-

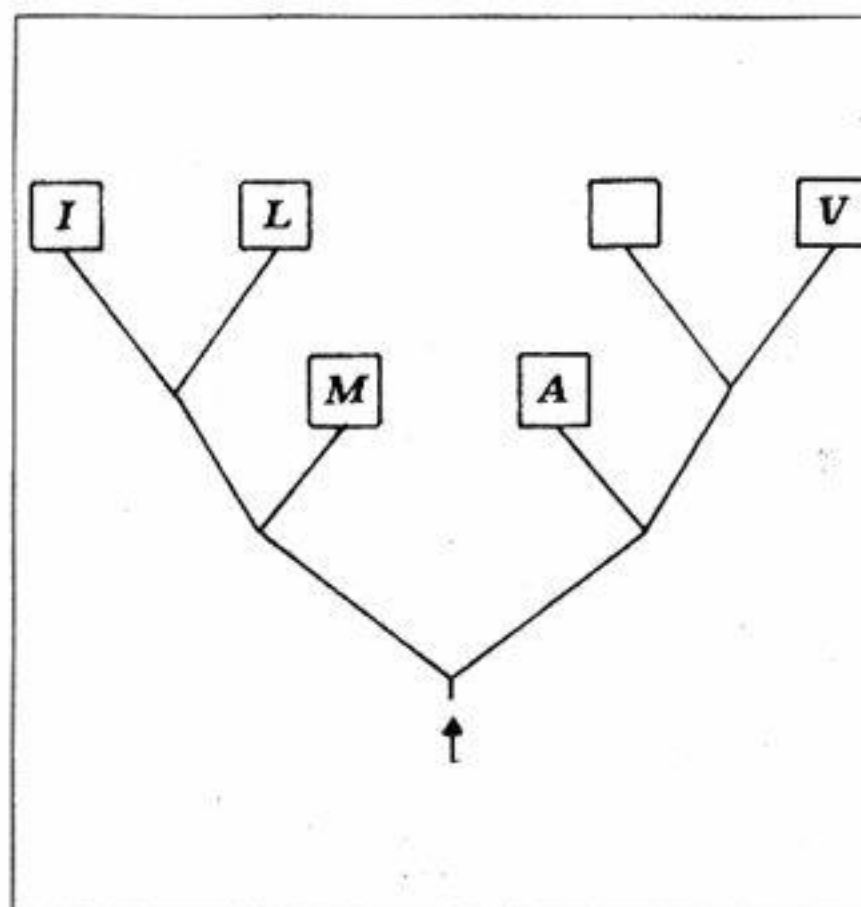
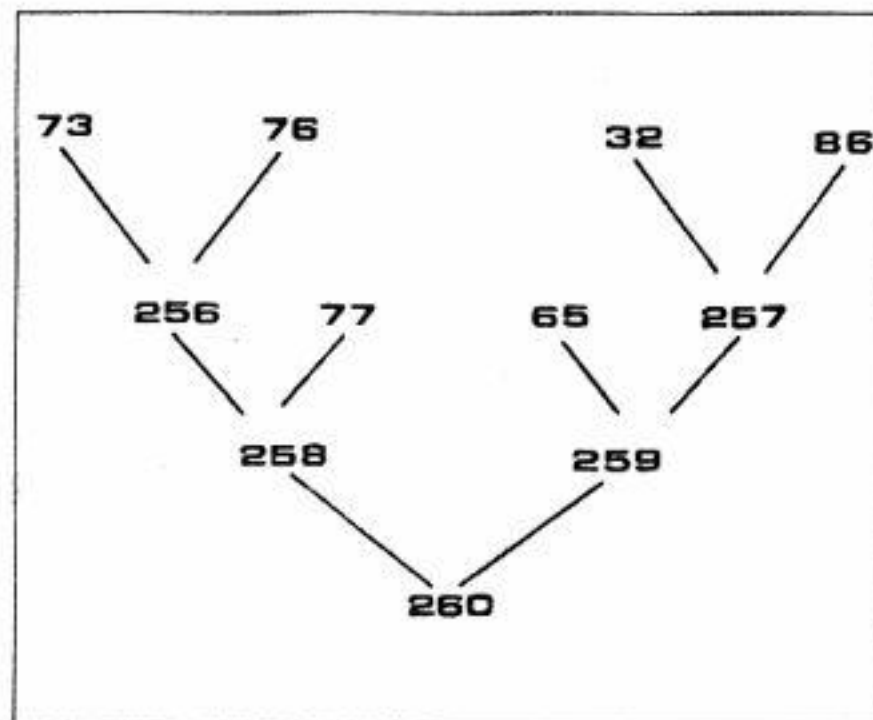
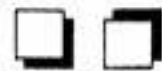



Figura 1
Un albero binario con informazione concentrata nei nodi terminali. In base a tale albero la lettera L (Codice Ascii 76) viene codificata con: **Codice$ (76) = "001"** in cui, partendo dalla freccia, **0** significa "gira a sinistra" e **1** significa "gira a destra; analogamente si avrà Codice$ (32) = "110" (per lo spazio), Codice$ (77) = "01" (per la lettera M) e così via.

> *A pensarci bene, non tutti i bit che costituiscono un file sono davvero "utili"*

COPIALO PER TELEFONO

Anche i listati presenti in queste pagine possono esser tirati giù per mezzo del modem; se, ovviamente, ne possedete uno.
La procedura per collegarsi con la nostra banca dati (attiva 24 ore su 24) è riportata su altra parte della rivista.
Chi non possiede il modem (che aspettate a procurarvelo?) può tuttavia richiedere il dischetto, contenente il software, presso il nostro servizio arretrati.

vere un po' di roba in Linguaggio Macchina; il riquadro specifico ("Il disassemblato") fornisce qualche spunto in proposito.

Decodifica del messaggio

Torniamo al messaggio "viva la mamma", ormai tradotto nei 4 byte 28, 46, 38, 105. Per "rileggere" tale messaggio, il vettore Codice$ non servirebbe assolutamente a nulla; farebbe invece comodo disporre di un vettore "Figlio", che esprima le "relazioni di parentela", come indicato in figura 2.

Una volta costruito tale vettore, la decodifica del messaggio è immediata: cominceremo a scomporre i 4 byte in 32 bit, riottenendo così la stringa B$; poi (si veda ancora la figura 2: 256 è il nome della radice, e i nodi terminali hanno nome < 256) faremo:

Messaggio$ = ""
For X = 1 to 13: padre = 256
For W = 1 to 999: Rem simula un Repeat...
B = Asc (B$) - 48: Rem B è la cifra binaria
B$ = Mid$ (B$, 2): Rem la prima cifra non serve più
Padre = Figlio (Padre, B)
If Padre > 255 Then Next: Rem... Until
Messaggio$ = Messaggio$ + Chr$ (Padre)
Next X: Rem questo chiude anche il ciclo in W

Naturalmente c'è un piccolo imbroglio: chi ci dice che il messaggio ha lunghezza 13? In realtà c'è un imbroglio ancora più grosso: chi ci ha dato il vettore Figlio?

La risposta è ovvia, anche se dolorosa: il mittente dovrà in qualche modo "trasmettere" al ricevente anche l'albero! E, visto che ci può servire, supporremo che il messaggio codificato contenga nell'ordine: la **lunghezza** del file originario, una **descrizione** dell'albero usato, e infine il **file** nella sua forma codificata.

Ancora due osservazioni: da un lato, nel caso di "veri" file, anche in fase di decodifica occorre inframmezzare le operazioni con il colloquio col disco, d'altro lato la "trasmissione dell'albero" non è gravosa: bastano, nel caso peggiore, 320 byte; se il messaggio è lungo, e se la compressione funziona bene, l'operazione sarà comunque conveniente. Quando l'albero di Huffman funziona bene? Essenzialmente nel caso di file con non troppi caratteri, presenti con frequenze abbastanza diverse; tipicamente i file **Ascii** e, in minor misura, **i listati Basic** (l'intero file compresso ha allora una lunghezza che oscilla tra il **30%** e il **40%** di quella originaria). Se al contrario, come avviene ad esempio per i programmi in Linguaggio Macchina, il file contiene tutti i 256 caratteri Ascii, con frequenze quasi uguali, si ha un guadagno del 15% - 25% che, se il file non è molto lungo, risulta poi quasi vanificato dal fatto che occorre preliminarmente trasmettere l'albero.

Conclusioni

Il primo listato pubblicato fornisce la decodifica di un file; la costruzione dell'albero di Huffman, ed il programma di codifica, è bene esaminarli solo **dopo** aver attentamente studiato quanto detto finora (comprese figure, riquadri e listato); gli impazienti potranno ottenere fin da ora il relativo listato (ovviamente anche in forma compressa: si può usare il listato di queste pagine per decomprimerla) presso la Banca Dati che la Systems ha recentemente istituito per i suoi lettori.

Per quanto riguarda il programma, esso **può girare indifferentemente su C/64, C/16, Plus/4 e C/128**; le routine in L.M. sono scritte sullo schermo, per evitare di occupare inutilmente altra memoria Ram. Grazie alle linee 180 - 220 il programma scopre da solo dove sta lo schermo (che è in locazioni diverse a seconda del computer usato) mettendosi addirittura in modo fast se si è su un C/128 con schermo a 80 colonne; la linea 1160, e la scrittura di certi dati con un asterisco (*) finale, permettono poi alle parti in L.M. di rilocarsi, su ogni Computer, nel punto giusto.

A proposito di altri trucchetti spiccioli, si osservino le linee 800 - 830 che consentono, a partire dal nome **FL$** di un file, di scoprire di che tipo è il file stesso (Seq, Prog o User; il trattamento di file di tipo **Rel** richiederebbe pesanti modifiche del Listato); e la subroutine 1040 che realizza un "Input guidato": all'utente viene posta una domanda (contenuta nella variabile **sh$**) evidenziata in reverse e seguita da una

Dalla mamma ai numeri

La figura 2b indica (tramite i rispettivi codici Ascii) le lettere presenti nel messaggio "Viva la mamma" e, tra parentesi, la relativa frequenza, cioè: v(2), i(1), a(4), spazio(2), l(1), m(3). Vedi schema in alto, qui a lato.

A partire da essa si eseguono le fasi (a) e (b):

(a) si cercano le due frequenze più basse; i caratteri corrispondenti (cioè **i**, **l**) vengono "dimenticati" e sostituiti con un carattere fittizio (per evitare confusioni coi veri codici Ascii partiremo dal 256) cui si attribuisce la frequenza somma.

(b) i caratteri soppressi vengono detti "figli" del nuovo carattere.

Il risultato dell'operazione è in figura 2b. Si ripete poi l'operazione stessa (senza distinzione tra caratteri "veri" e caratteri fittizi); altre due operazioni portano alla figura 2c; due ancora portano alla figura 2d, che è una descrizione equivalente dell'albero di figura 1.

"risposta plausibile" (contenuta nella variabile **ri$**); se l'utente è d'accordo può limitarsi a premere Return, sennò scriverà la risposta desiderata (per motivi ovvi, se il programma si trova a dover sovrascrivere un file, la risposta suggerita è "no").

Per il resto, alcune Rem sparse qua e là dovrebbero facilitare la comprensione del listato; la lettura dell'albero (effettuata dalla subroutine 550) è un problema delicato che, come il problema gemello relativo alla scrittura dell'albero, va risolto ricorsivamente; a tali problemi sono dedicati gli ultimi riquadri, riservati solo ai lettori più esperti.

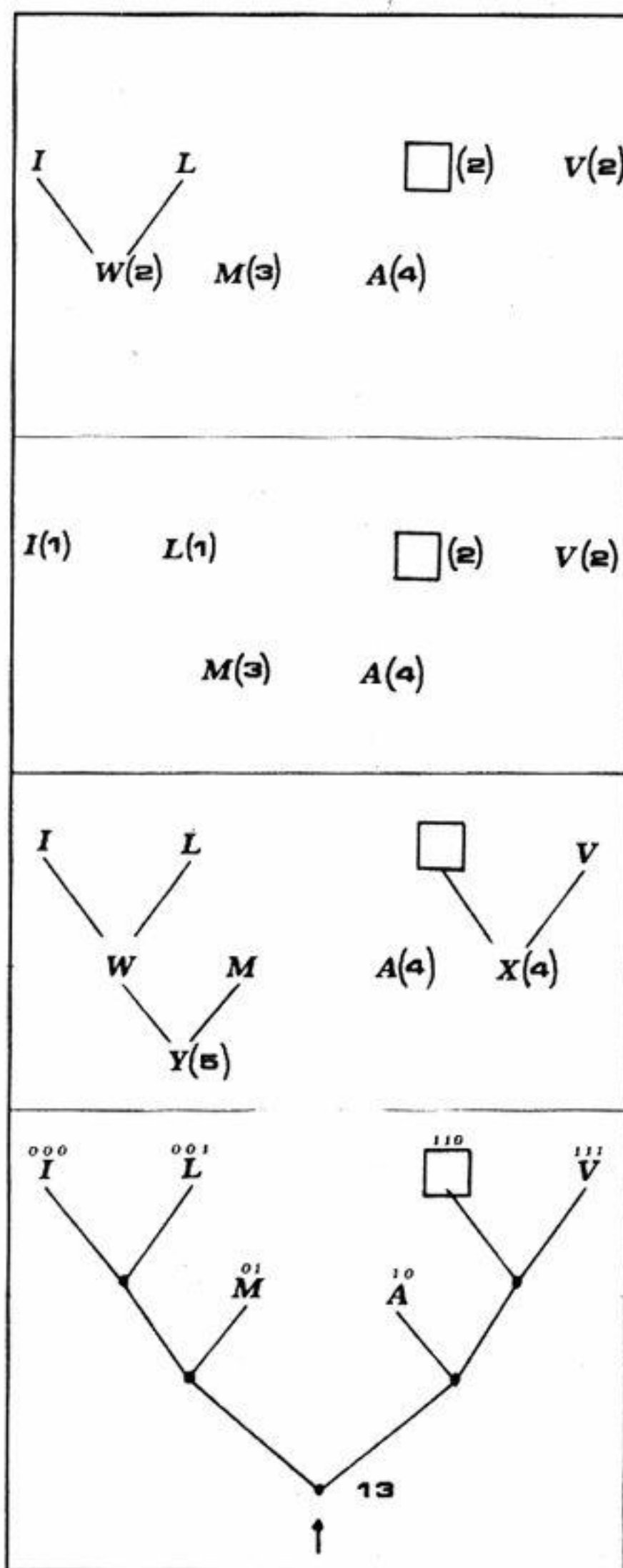
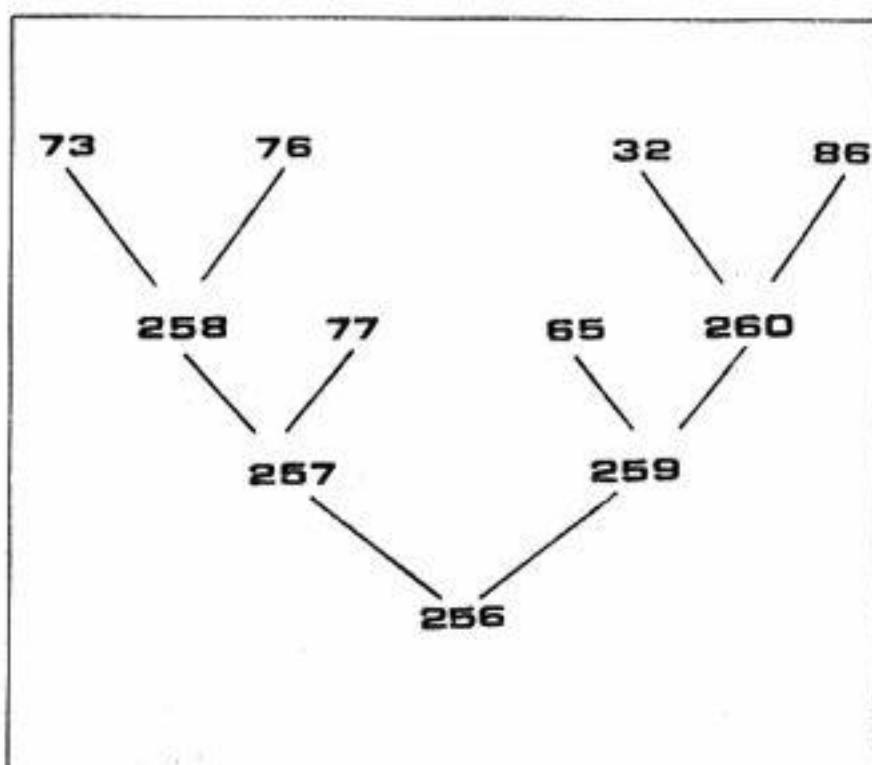



Figura 2- a/b/c/d
Sviluppo dell'albero di Huffman del file "viva la mamma": in **a** sono descritti i caratteri presenti e le loro frequenze; in **b** i due caratteri meno frequenti (la I e la L) sono divenuti "figli" di un carattere fittizio (W, il primo di quelli che non appaiono nel file) cui è attribuita la frequenza somma delle frequenze dei suoi figli; altre due operazioni portano alla **c**; altre due ancora completano l'albero (**d**; la frequenza della radice è la lunghezza totale del file). I codici srcitti sopra le lettere **d** rappresentano la strada che, partendo dalla radice, porta alla lettera in questione (0 = volta a sinistra; 1 = volta a destra).

La procedura descritta è valida (ed automatica) qualunque sia il "piccolo" Commodore che possedete: C/64, C/16, Plus/4, C/128

```
.678   ldx #$08       ; canale del file #8:
.67a   jsr $ffc6      ; stai pronto a mandare dati
.67d   ldx #$00       ; contatore
.67f   txa            ; valore 0
.680   sta $0400, x   ; in pagina 4
.683   inx            ; ripeti fino a riempire
.684   bne $0680      ; la pagina di valori 0
.686   jsr $ffa5      ; disco mandami un dato
.689   ldy #$08       ; contabit
.68b   rol            ; se il bit alto è 0
.68c   bcc $0691      ; va bene valore 0
.68e   inc $0400, x   ; sennò valore 1
.691   inx            ; posizione vicina
.692   dey            ; next bit
.693   bne $068b      ; fino a farne 8
.695   lda $90        ; leggi la variabile st
.697   cmp #$40       ; se vale 64
.699   beq $069f      ; smetti, era l'ultimo del file
.69b   cpx #$00       ; se non hai fatto 32 byte
.69d   bne $0686      ; fanne ancora uno
.69f   jmp $ffcc      ; canali chiusi e rientro.
```

Il disassemlato

Una volta aperto da Basic un file in lettura (ad esempio, con il numero 8), la lettura del file da L.M. è semplicissima: le istruzioni contenute nelle locazioni 678, 67A (vedi disassemblato) avvertono il disco di tenersi pronto ad inviare i dati; la 686 riceve un dato dal disco, ponendolo in Acc. Quando il dato letto è l'ultimo del file, nella locazione **$90** (corrispondente alla variabile riservata **ST** del Basic) si troverà il valore **64**.

Letti i dati voluti, bisogna ricordarsi di avvisare il disco (tramite la 69F) che il colloquio è finito. La chiamata **SYS 1656** (= $678) legge dal disco un blocco di 32 byte (o meno, se il file finisce prima) e ne sparpaglia la scrittura binaria sulle prime 256 locazioni di schermo.

Codifichiamo

Riprendendo il discorso sulla compressione, in questa seconda parte costruiremo il programma di **codifica** da abbinare a quello di **decodifica** presentato nelle pagine precedenti; la costruzione in Basic dell'albero di Huffman sarà ottenuta tramite una strategia di tipo **Heap-Sort**, il che ci darà modo di occuparci di una delle più efficienti strategie di riordinamento finora proposte.

Mettiamo in chiaro

Mettiamoci nei panni del decodificatore che riceve una stringa binaria del tipo della Albero$ (definita nel riquadro "Di padre in figlio", e ridotta a 58 bit) e deve ricostruire l'albero di figura 2 (vedi figura 3).

Nel listato è la subroutine 570 che si occupa del lavoro; per capire che cosa fa, conviene sfrondarla dalle coversioni binario / decimale.

Battezziamo la radice con il numero 256; e usiamo una variabile **NF** (Next Free) che fornirà il numero di un nodo non ancora utilizzato; come nel riquadro citato, la variabile **liv** misura il "livello di ricorsione" ed il vettore **H** servirà per memorizzare dei dati nel corso delle operazioni.

La costruzione si sviluppa in due fasi (quasi identiche tra loro): si tratta prima il Ramo Sinistro **RS** (linee 2000 - 2030), poi il Ramo Destro **RD** (linee 2040 - 2070) del sotto albero che parte dal nodo h (liv):

2000 padre = h (liv): (leggi un bit b)
2010 if b = 1 then (leggi un byte a): figlio (padre, 0) = a: goto 2040
2020 figlio (padre, 0) = nf: liv = liv + 1: h (liv) = nf: nf = nf + 1
2030 gosub 2000: liv = liv-1: padre = h (liv)
2040 padre = h (liv): (leggi un bit b)
2050 if b = 1 then (leggi un byte a): figlio (padre, 1) = a: return
2060 figlio (padre, 1) = nf: liv = liv + 1: h (liv) = nf: nf = nf + 1
2070 gosub 2000: liv = liv-1: return

L'esecuzione di *root = 256: nf = root + 1: liv = 0: h (liv) = root: gosub 2010* fornirà l'intero albero.

Si noti che tali operazioni NON forniscono l'albero di figura 2, bensì quello di figura 3, nel quale la numerazione dei nodi non terminali è diversa; ciò tuttavia non influisce minimamente nelle successive operazioni di decodifica.

Di padre in figlio

Stabiliamo di chiamare nodi di **tipo 0** i punti di biforcazione (che sono a loro volta la radice dei due sotto - alberi **RS**, ramo sinistro e **RD**, ramo destro); e di **tipo 1** i nodi terminali.

Se accettiamo di chiamare albero anche un oggetto *degenere* costituito da un solo nodo terminale, abbiamo pronta per l'uso una (quasi rigorosa) definizione ricorsiva di albero (meglio: del tipo di albero che qui ci interessa):

Un albero è: un **1** seguito da un *carattere*, oppure uno **0** seguito da *RS, RD*; dove a loro volta RS e RD sono alberi.

Passata la sorpresa, osserviamo che, con questa interpretazione, la figura 2 può essere descritta dalla stringa:

Albero$ = "0001i1l1m01a01 1v"

dove appunto (in conformità con i due tipi di nodi) uno **0** significa "siamo in una biforcazione, segue la descrizione di RS, poi quella di RD"; mentre un **1** significa "siamo in un nodo terminale, segue l'indicazione del carattere". Naturalmente, per fare le cose per bene, dovremo sostituire ogni carattere con il suo codice Ascii, o meglio con la scrittura binaria di tale codice; la "vera" stringa Albero$ ha perciò lunghezza 59 (in bit; poco più di 7 byte). E' tale stringa che trasmetteremo per descrivere l'albero; anzi, per facilitare l'operazione di decodifica (si veda il riquadro 3), sopprimeremo lo 0 iniziale che corrisponde alla radice dell'albero.

Avendo a disposizione il vettore Figlio come indicato in figura 2 (Root indicherà la radice, nel nostro caso 260), e sfrondando il discorso dalle noiose (ma necessarie!) conversioni da 8bit ad un byte, basterà scrivere:

```
3000 if padre < 256 then 3040
3010 print "0";
3020 h (liv) = padre: liv = liv + 1: padre = figlio (padre, 0): gosub 3000
3030 padre = figlio (h (liv-1), 1): gosub 3000: liv = liv-1: return
3040 print "1";: print (scrittura binaria di padre);: return
```

e l'intera operazione sarà realizzata da:

```
liv = 0: padre = root: gosub 3020
```

Per interpretare queste poche righe, si tenga presente che la variabile **liv** indica il "livello di ricorsione" (in soldoni: quanto ci siamo allontanati dalla radice), e che in generale la chiamata **gosub 3000** effettua la scrittura del sotto albero che ha per radice il nodo "padre". A cosa serve la posizione *h (liv) = padre* in riga 3020? E' presto detto: ogni volta che si incontra un nodo non terminale (padre > 255) occorre trasmettere uno **0** seguito dai rami sinistro e destro che partono da tale nodo. Per trasmettere i due rami si farà *gosub 3000* con padre sostituito prima dal suo figlio di sinistra, poi da quello di destra.

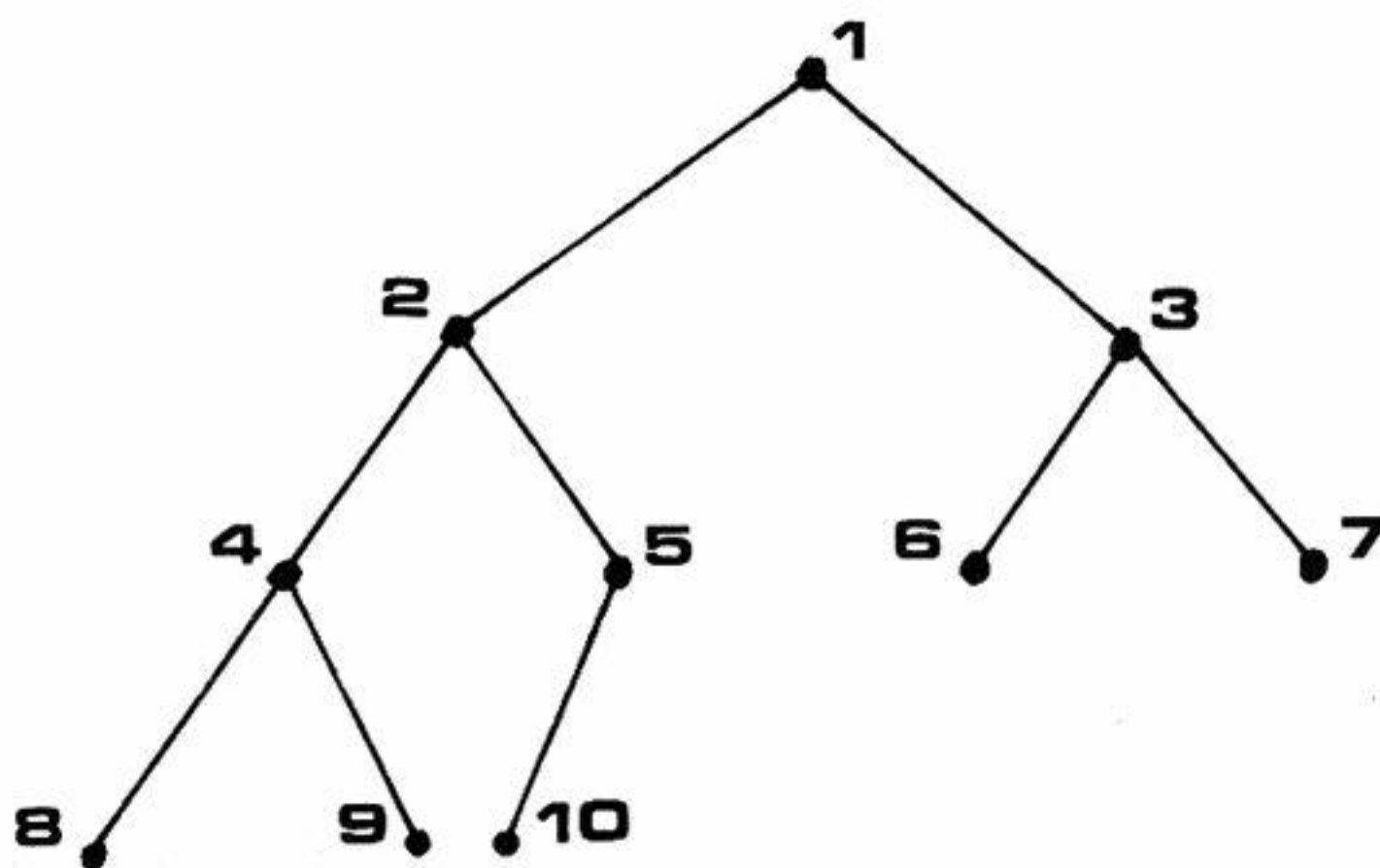
In linguaggi evoluti come il Pascal ciò non è un problema; in Basic, invece, occorre memorizzare il valore di padre con cui si sta lavorando, per poterlo poi riutilizzare.

Chi possiede un Amiga può tentare di implementarvi la procedura descritta dal momento che questa è di tipo "universale"

Figura 3

Inutile dire che i listati devono esser digitati con la massima attenzione

Preparazione dell'albero

Riepiloghiamo come si fa a costruire l'albero di Huffman di un file; occorre:

(a) scandire il file per individuare quali caratteri (codici Ascii da 0 a 255) sono presenti in esso; per ogni x tra 0 e 255 si deve contare quante volte il carattere Chr$ (x) compare nel file;

(b) individuare il carattere che compare più di rado (o fissarne uno, se ci sono degli ex-aequo) e chiamarlo *figlio di sinistra* di un carattere fittizio;

(c) individuare il successivo carattere con frequenza minima e chiamarlo *figlio di destra* del carattere fittizio;

(d) sopprimere i due caratteri scelti in (b) e (c), ed assegnare al carattere padre una frequenza pari alla somma dei caratteri figli;

(e) ripetere le fasi (b), (c), (d), creando ogni volta un nuovo carattere fittizio; nelle fasi (b) e (c) non si fa distinzione tra i caratteri *veri* rimasti e quelli *fittizi* già creati.

Poichè, ad ogni iterazione, scompaiono due caratteri e ne nasce uno nuovo, se nel file sono presenti **NT** caratteri distinti variamente ripetuti, dopo **NT-1** iterazioni il processo avrà termine, e l'albero sarà completato (si veda la figura 1 che è relativa ad un file il cui contenuto è "viva la mamma").

Naturalmente la fase (a) sarà in parte sviluppata in Linguaggio Macchina (linea 460 del listato); se, durante tale fase, valutiamo anche quale è il codice Ascii più alto che compare nel file (diciamo **Max** tale valore, che è al più 255), per i "caratteri" che dobbiamo inventarci sceglieremo i valori Max + 1, Max + 2,..., Max + NT-1; per memorizzare le frequenze useremo un vettore **Fr** (sta per Frequenza; se lo chiamate proprio "Frequenza", non arrabbiatevi di fronte al Syntax Error dovuto al *Fre...*).

Durante l'iterazione delle fasi (b) e (c) converrà memorizzare le relazioni di parentela via via costruite tramite **Due** vettori, Figlio e Padre (nel seguito abbreviati **Fi** e **Pa**; prima, lo ricordate?, era sufficiente il solo Fi): Fi (x, 0), Fi (x, 1) e Pa (x) forniranno rispettivamente il figlio di sinistra, il figlio di destra, ed il padre del "carattere" x (naturalmente i caratteri "veri" non hanno figli, e la "radice" dell'albero non ha padre). Si tratta di informazioni ridondanti, ed apparentemente inutili; al codificatore serve (come visto la volta scorsa) un altro tipo di informazione: per ogni carattere x presente nel file occorre costruire una stringa Codice$ (x), composta da zeri e uni, che fornisce l'itinerario per arrivare a tale carattere: partendo dalla radice, uno zero significa "volta a sinistra", un uno significa "volta a destra".

In realtà, per la costruzione del vettore Co$, fanno comodo entrambi i vettori Pa e Fi (linee 790 - 830; si noti l'uso di un ciclo For... Next nella variabile y, che rimpiazza la costruzione Do... Loop... Until, assente nel Basic 2.0 del C/64; i possessori di C/16 inespanso (poveracci...), oltre a dichiarare di tipo % i vettori Fi e Pa, potrebbero memorizzare su un file disco i valori di Fi e di Co$, spezzando qui il programma; una seconda parte (nella quale i vettori Pa e Fr non servono più) si occuperà di rileggere tali valori e completare i lavori.

Costruzione del file compresso

Salvo il dettaglio delle fasi (b) e (c), dettaglio di cui ci occuperemo tra poco, ormai tutto è pronto per costruire il file compresso; ricordiamo che avevamo stabilito che in tale file vanno scritti nell'ordine: la **lunghezza** del file originario (che è la frequenza della radice); una **descrizione** dell'albero (che non staremo a ripetere; è in tale fase, sviluppata ricorsivamente nella subroutine 1060 - 1130 che serve ancora il vettore Fi); e infine il "vero" **file**: tramite una *seconda scansione* del file da comprimere, per ogni carattere x che leggiamo dal file dovremo accodare al file in costruzione la stringa Co$ (x). L'ultimo passaggio è il più scocciante, anche se banale: la stringa di zeri e uni, che via via si allunga, deve ogni tanto essere "scorciata": i suoi primi 8 caratteri vengono soppressi ed è il byte da essi rappresentato che va inviato al disco. Nel secondo listato, alcune di queste fasi (quelle da ripetere più volte, e quelle che risulterebbero troppo lente in Basic) sono sviluppate in Linguaggio Macchina; in particolare, per ogni carattere x letto dal file si esegue un "Print Co$ (x);" e quando si sono scritti 256 caratteri, una routine in L.M. si occupa di leggerli dallo schermo, compattarli in 32 byte, e inviarli al disco.

Volendo scrivere un programma che potesse girare su ogni modello Commodore (C/64, C/128, C/16 e Plus/4) ci si imbatte in qualche problema di compatibilità: le routine in L.M. fanno uso solo dei vettori del Kernal, quindi vanno bene su ogni modello; ma dove scriverle? Essendo molto corte, esse troverebbero posto ovunque; ma usano 512 locazioni per il colloquio col disco; dove trovare tale spazio? Sul C/64 si potrebbe usare la Ram nascosta (da $C000 in su) che però non esiste sul C/16, mentre su C/128 e Plus/4 dovrebbe essere "protetta" abbassando il top delle stringhe (top definito da puntatori diversi sulle due macchine); sul C/128 nascono anche problemi di Banchi, e sul C/16 inespanso ogni locazione rubata

VIETATO AI MINORI

E' probabile che i neo-utenti di computer non riescano a ben comprendere l'utilità della procedura software descritta nelle presenti pagine. Questa è infatti destinata a coloro che sono già padroni delle principali tecniche di programmazione e desiderano approfondire il modo in cui un elaboratore organizza i vari dati all'interno della memoria.
Il programma, pertanto, rappresenta un invito ai lettori più esperti, soprattutto a coloro che, considerando il listato come una base di partenza, riescano a pervenire a procedure più interessanti e capaci, magari, di offrire applicazioni di più ampio respiro.
Se hai incominciato da poco, comunque, non scoraggiarti! Tutti coloro che, oggi, vantano una particolare competenza nel campo dell'informatica hanno iniziato con un banale Print "Pippo". Perchè non dovresti riuscire anche tu?...

al Basic rischia di generare un "out of memory"...

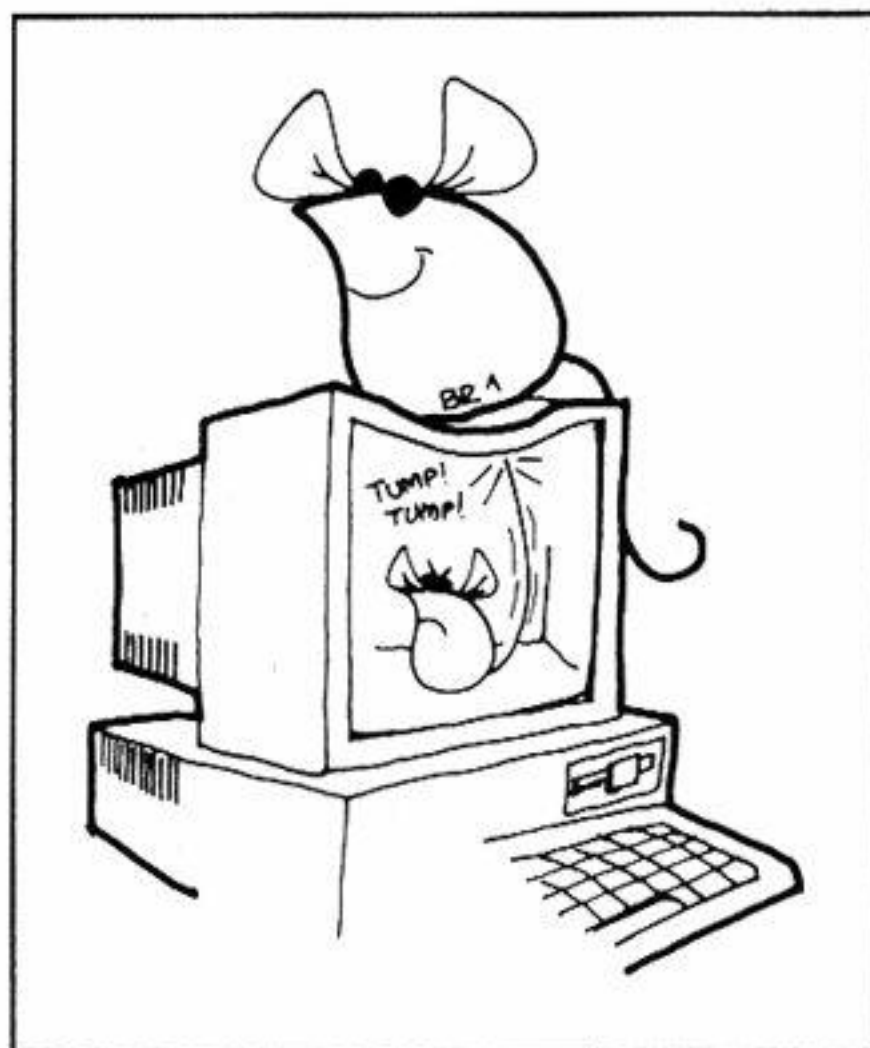
La soluzione adottata, sia pure al prezzo di qualche complicazione in più nel listato, fa piazza pulita di tutti questi problemi: le ruotine in L.M sono scritte sullo schermo, e pure sullo schermo trovano posto i dati che il L.M. trasferisce dal/al disco. E' vero che il valore **BS** (Base Schermo) è diverso nei vari modelli (**1024** per C/64 e C/128, **3072** per gli altri); ma tale valore il programma se lo calcola facilmente (linee 210 - 230), e poi si adegua. Sul C/128 ciò costringe a lavorare con schermo a 40 colonne e quindi in modo slow; chi, avendo il Monitor adatto, avrebbe preferito lavorare in modo fast, sappia che gli conviene senz'altro compilare il programma (con Austro Speed; il Pet Speed maltratta i programmi che usano la ricorsione) e lavorare in modo C64.

L'albero di Huffman

Abbiamo finora rinviato la discussione di come realizzare le fasi (b) e (c) per la costruzione dell'albero. Si tratta di un problema banalmente risolubile (chi non è in grado di scrivere due righe per trovare il valore minimo di un vettore? Se si hanno N elementi bastano N-1 confronti!); eppure è concettualmente il più interessante, perchè si presta ad essere sviluppato in vari modi; e la scelta del metodo utilizzato ha un'influenza sensibile sull'efficienza del programma; cerchiamo di capire perchè.

Detto **NT** il Numero Totale dei codici Ascii presenti almeno una volta nel file, lo sviluppo della fase (b) richiederà NT-1 confronti; per la fase (c) ne serviranno NT-2; la fase (d) riporta il numero di caratteri a NT-1. Sorvolando su qualche dettaglio (che appesantisce però il lavoro da eseguire!) la prima esecuzione delle fasi (b), (c), (d) "costa" 2NT - 1 confronti. Poichè ora NT è calato di 1, la seconda iterazione costerà 2 (NT - 1) -1 confronti, cioè 2NT - 3; la terza costerà 2NT - 5 e così via. Un conto banale mostra che in tutto saremo costretti a fare NT^2 confronti. Si dice allora che il metodo è "quadratico"; che cosa significa in pratica? Se ad es., con NT = 10, il nostro programma impiega 5 secondi a costruire l'albero, con NT = 20 ne impiegherà 20 (il quadruplo); con NT = 40 impiegherà più di un minuto (ancora il quadruplo, 80 secondi) e così via.

In generale: raddoppiando NT, il tempo quadruplica. Sarebbe molto più utile trovare una strategia "lineare" anzichè "quadratica"; magari più lenta sui piccoli numeri (diciamo che impieghi, ad esempio, mezzo minuto anzichè 5 secondi per NT = 10) ma che poi, se si raddoppia NT, il tempo di esecuzione si limiti a raddoppiare. In realtà strategie veramente lineari sono impossibili; vedremo però che esiste una strategia "quasi lineare", che impiega un tempo proporzionale a NT * Log (NT) (per chi abbia scordato i logaritmi: il guadagno rispetto alla prestazione quadratica è notevole).

Una soluzione che sembra plausibile consiste nel costruirsi preliminarmente una "classifica" dei vari caratteri; cioè un vettore CI tale che CI (1) sia il carattere con frequenza più bassa, CI (2) il successivo, e così via fino a CI (NT) che è quello a frequenza massima. Si tratta, in sostanza, di un'operazione di riordinamento; ed

esistono strategie efficienti per fare ciò ad es. quella presentata nel listato del riquadro. A questo punto le fasi (b) e (c) sono banali (i caratteri da eliminare sono CI (1) e CI (2)); e la fase (d) richiede solo di mettere al giusto posto nella classifica il carattere appena creato...; in realtà qui crolla tutto. Trovare il giusto posto per un nuovo elemento in un insieme ordinato di N elementi "costa" solo Log (N) confonti; ma poi "sistemare" l'elemento al posto giusto richiede mediamente N/2 operazioni (di tipo scambio); il procedimento, a conti fatti, risulterà ancora quadratico.

❑❒

Uno schema di tipo Heap - Sort

L'idea è che in realtà a noi non serve affatto costruire una classifica dei caratteri in base alla loro frequenza: ci basta disporli secondo una struttura che renda facili (cioè rapide) tre operazioni:

(1) trovare l'elemento più piccolo (meno frequente);

(2) sopprimere dalla struttura un elemento ormai inutile;

(3) inserire un nuovo elemento nella struttura;

Rispetto alla struttura di ordine la (1) è banale; la (2) è facile perchè, nel nostro caso, l'elemento da sopprimere è sempre il più piccolo; e la (3) è invece, come si è visto, un'operazione lenta. Una struttura migliore, che anzi sembra tagliata su misura per queste tre operazioni, è la cosiddetta struttura di **Heap**, illustrata nel riquadro corrispondente.

Ovviamente, in uno Heap, la ricerca dell'elemento minimo è banale; come si fa (e quanto costa) un'operazione di "inserimento" in uno Heap di N elementi? L'operazione è semplicissima: basta inserire l'elemento al primo posto libero (l'N + 1-esimo), poi confrontarlo (ed eventualmente scambiarlo) con suo "padre" fino a che non è risalito al posto giusto. Il numero massimo di confronti e scambi è perciò dato dalla "profondità" dell'albero, cioè è dell'ordine di Log (N) (si veda ancora la didascalia di figura 2).

Analogamente si controlla che è facile, e costa ancora un tempo proporzionale a Log (N), la soppressione di un elemento del mucchio; e che la riorganizzazione a forma di Heap di un generico insieme di N elementi ha un costo dell'ordine di N * Log (N).

A conti fatti, quando applicata alla costruzione dell'albero di Huffman, la strategia di Heap-Sort richiede un tempo totale dell'ordine di NT * Log (NT).

Il programma di Heap

Nel listato le righe 100 - 270 costituiscono un demo; le righe da 50000 in poi sono scritte in forma di subroutine, per poterle "accodare" ai propri programmi: l'uso costante di cicli For.. Next per simulare i Repeat.. Until (in modo da evitare dei lenti "Goto all'indietro") fa sì che i tempi di esecuzione non dipendono da quanto è lungo il resto del programma.

Un esempio di tempi su C/128 in modo Fast: 25, 50, 100, 200 elementi sono riordinati rispettivamente in 3, 8, 18, 42 secondi. Si tratta di tempi di tutto rispetto, nettamente inferiori a quelli ottenibili con metodi più tradizionali e (ingiustamente) più famosi.

In realtà il metodo detto *Quick - Sort*, se programmato bene (evitando, cioè, i Goto all'indietro), viaggia mediamente a velocità doppia; però, se applicato a vettori "già quasi in ordine", risulta disastrosamente lento.

Nel listato, oltre alla subroutine 51000, commentata nelle Rem che la precedono, si noti che la linea 50010 crea uno Heap per A$ (1),..., A$ (N). La nozione di Heap è presentata nel riquadro conclusivo; qui lo Heap è alla rovescia: gli elementi padri sono maggiori degli elementi figli.

Fatto ciò, se N = 2, basta scambiare i due elementi (linea 50050); mentre, se N > 2, si iterano le operazioni: scambia primo ed ultimo elemento (così l'elemento più grande va in coda); poi poni N = N-1 (così l'elemento più grande non sarà più toccato); infine ripristina lo Heap (che ha solo il primo elemento fuori posto).

La linea 49980 (che non sarà mai elaborata!) è un modo, "resistente ai Renumber", per ricordarsi dove vanno apportati i cambiamenti qualora il vettore da riordinare, invece che A$, si chiamasse in un altro modo, o avesse una struttura diversa (ad es.: se si tratta di rappresentazioni di numeri in forma di stringhe il confronto va fatto su Val (A$ (B% (.)) anzichè su A$ (B% (.)); ovvero, se A$ è "a più dimensioni", si deve indicare rispetto a quale componente riordinare; eccetera).

Alberi e logaritmi

I numeri da 1 a N, con N = 10, sono stati disposti ad albero binario. Il nodo numero K non ha figli se 2K > N; ha esattamente due figli (i nodi 2K e 2K + 1) se 2K < N; e ne ha uno solo se 2K = N.

Si pensi ad un albero genealogico costituito da N persone, ognuna delle quali abbia al più due figli; e si costruisca il vettore D$ ponendo D$ (K) = data di nascita di chi è inserito nel nodo numero K.

Chiaramente D$ NON è un vettore ordinato (uno zio potrebbe essere nato dopo un nipote; e le età dei cugini possono essere nel disordine più totale); tuttavia si tratta di una "struttura di Heap" nel senso che la data di nascita di ogni figlio è maggiore di quella del genitore.

In generale un vettore A (1), A (2),..., A (N) (poco importa se gli elementi sono letterali o numerici) si dice uno Heap se per ogni K il valore di A (K) è al più uguale al valore di A in ognuno degli eventuali (al massimo due) figli di K. La traduzione letterale di **Heap** sarebbe **Mucchio**; ma il termine è così poco espressivo che conviene abituarsi ad usare la parola inglese.

La "profondità" dell'albero, cioè la massima lunghezza dei cammini che partono dal capostipite ed arrivano ai vari discendenti (o ancora: il numero di generezioni rappresentate nell'albero) cresce molto lentamente al crescere di N: se N raddoppia essa si limita ad aumentare di 1; matematicamente ciò si traduce dicendo che è proporzionale al Logaritmo di N.

CONCLUSIONI

Nel listato abbiamo evidenziato chiaramente le varie fasi: le 600 - 610 creano lo Heap, le 650 - 680 creano l'albero, le 790 - 830 creano i codici e valutano il guadagno realizzato dalla compressione. Si noterà che, nelle linee 600, 650, 800 è inserito il comando **Print ".";** . Si tratta di un fronzolo per chi, come me, soffre di "schermo-vuoto-fobia": il procedimento è ormai abbastanza veloce, e rallentarlo un pochino con tali comandi non costa molto; naturalmente chi non soffre di certe fobie può tranquillamente sopprimere tali comandi di Print.

L'uso di una struttura di Heap può rivelarsi utilissimo in ogni situazione in cui, come nel caso dell'albero di Huffman, non serva veramente un ordine, ma solo un rapido accesso agli elementi più piccoli; ad es. se si deve gestire una "coda" in cui ogni evento in attesa abbia una sua "priorità". Per altro, appoggiandosi alla nozione di heap, è possibile costruire una delle più efficienti strategie di riordinamento, come illustrato nel riquadro specifico.

Nei casi più fortunati la compressione può raggiungere il 50% della lunghezza originaria del file

```
100 rem" ┌──────────────────────────────┐
110 rem" │  HUFFMAN : Decompressione    │
120 rem" └──────────────────────────────┘
130 rem
140 rem" ┌──────────────────────────────┐
150 rem" │       C.Baiocchi; V.1990     │
160 rem" └──────────────────────────────┘
170 rem
180 bs=1024:printchr$(14)"sss";:ifpeek(bs)<>32thenbs=3072
190 print"a":ifpeek(bs)=1then230
200 bs=1024:printchr$(27)"xsss";:ifpeek(bs)<>32then220
210 print"a"chr$(27)"x":ifpeek(bs)=1then=%:goto230
220 print"sNon trovo lo schermo !":end
230 print"sDecompressione":print"Il programma :
240 print"i)   NON accetta file di tipo REL.
250 print"ii)  Lavora su UN SOLO disco su cui,
260 print"     oltre al file da leggere, ci sia
270 print"     spazio per il file da scrivere.
280 print"iii) Usa lo schermo per rielaborare le
290 print"     informazioni via via acquisite.
300 print"     se si preme lo STOP, poi si deve
310 print"     RICOMINCIARE DA CAPO !!!":print
320 dim fi(511,1),h(63)
```

```
330 show$=" # device":rispsta$="8":gosub1040:dn=val(ri$)
340 print:print"inserisci il disco voluto e premi *"
350 geta$:ifa$<>"*"then350
360 open15,dn,15,"i":s=st>0
370 input#15,e,e$:s=sor(st<>64):ifeorsthen1000
380 gosub780:rem chiede nomi per i file e li apre (# 8 in lett., #9 in scritt.)
390 gosub450:close15:end
400 :
410 rem"  ┌─────────────────┐
420 rem"  | Decompressione  |
430 rem"  └─────────────────┘
440 :
450 gosub1150:rem" routine su schermo in L.M.;sys sy legge 32 byte da disco
460 rem e li scompatta in 256 bit a partire da bs
470 :
480 rem lettura albero
490 :
500 print"sqqqqqqqqqqqqqqqqqqq"
510 print"Tutto OK, leggo l'albero
520 input#8,lun:root=256:li=0:h(li)=ro:nf=ro+1
530 fine=bs+256:pl=fi-1:rem pl = puntatore lettura; ho gia' letto peek(pl)
540 gosub550:goto690
550 pa=h(li):pl=pl+1:ifpl=fithensyssy:pl=bs
560 ifpeek(pl)=0then590
570 a=0:fory=1to8:pl=pl+1:ifpl=fithensyssy:pl=bs
580 a=a+a+peek(pl):next:fi(pa,0)=a:goto600
590 fi(pa,0)=nf:li=li+1:h(li)=nf:nf=nf+1:gosub550:li=li-1:pa=h(li)
600 pl=pl+1:ifpl=fithensyssy:pl=bs
610 ifpeek(pl)=0then640
620 a=0:fory=1to8:pl=pl+1:ifpl=fithensyssy:pl=bs
630 a=a+a+peek(pl):next:fi(pa,1)=a:return
640 fi(pa,1)=nf:li=li+1:h(li)=nf:nf=nf+1
650 gosub550:li=li-1:pa=h(li):return
660 :
670 rem lettura del testo
680 :
690 print"sqqqqqqqqqqqqqqqqqqq"
700 print"L'albero e' o.k.; traduco il testo
710 forx=1tolu:pa=ro:fory=1tole9:pl=pl+1:ifpl=fithensyssy:pl=bs
720 pa=fi(pa,peek(pl)):ifpa>rothennext
730 print#9,chr$(pa);:nextx:close8:close9:print"sFinito":return
740 :
750 :
760 rem" nome file, ricerca del tipo; su C128,Plus4,C16 premettere: directory
770 :
780 sh$="file da decomprimere":ri$="":gosub1040
790 fl$=ri$:l=len(fl$):ifl=0then780
800 ty$=",s":gosub850
810 ife=64thenclose8:ty$=",p":gosub850
820 ife=64thenclose8:ty$=",u":gosub850
830 ifethen1000:rem 64 = file type mismatch
840 print"un nome per il file decompresso":inputfs$:goto910
850 open8,dn,8,fl$+ty$+",r":input#15,e,e$:return
860 :
870 rem file su cui scrivere
880 :
890 print"Dammi un altro nome per il file
900 sh$="(senza virgole, *, ?, e simili)":ri$="":gosub1040:fs$=ri$
910 nome$=fs$+ty$+",w":open9,dn,9,nome$:input#15,e,e$
920 ife=0thenreturn
930 close9:if(e<>63)then1000:rem 63 significa file exists
```

```
940 print"Sul disco c'e' gia' un file col nome ▀"fs$
950 sh$="lo sopprimo":ri$="no":gosub1040:ifri$<"s"then890
960 a$="s"+d$+":"+fs$:print#15,a$:goto910
970 :
980 rem problemi col disco
990 :
1000 print"probl. col disco : "e$:close15:clr:end
1010 :
1020 rem input guidato
1030 :
1040 print:printspc(len(sh$)+4)ri$"▀":print" ▀"sh$;
1050 input"▀ ";ri$:return
1060 :
1070 data632:rem offset=$278; sys bs+$278 legge 32 byte (o meno se st=64)
1080 rem e li scompatta scrivendo i 256 bit a partire da bs.
1090 :
1100 data162,8,32,198,255,162,0,138,157,0,4*,232,208,250,32,165
1110 data255,160,8,42,144,3,254,0,4*,232,136,208,246,165,144,201
1120 data64,240,4,224,0,208,231,76,204,255
1130 data-1
1140 :
1150 print"▀":reada:sy=bs+a:forx=sytole9:reada$:ifa$="-1"thenreturn
1160 a=val(a$):ifright$(a$,1)="*"thena=a+(bs-1024)/256
1170 pokex,a:next
1180 end

ready.

100 rem" ┌──────────────────────────────┐
110 rem" |   HUFFMAN : Compressione     |
120 rem" ├──────────────────────────────┤
130 rem" | Su C128 si DEVE lavorare     |
140 rem" | su schermo a 40 colonne      |
150 rem" └──────────────────────────────┘
160 rem
170 rem" ┌──────────────────────────────┐
180 rem" |     C.Baiocchi; V.1990       |
190 rem" └──────────────────────────────┘
200 rem
210 bs=1024:printchr$(14)"▀▀▀";:ifpeek(bs)<>32thenbs=3072
220 print"a":ifpeek(bs)=1then240
230 bs=1024:print"▀Cambia schermo !":wait0:-&:run
240 print"▀Compressione":print"Il programma :
250 print"i)   NON accetta file di tipo REL.
260 print"ii)  Lavora su UN SOLO disco su cui,
270 print"     oltre al file da leggere, ci sia
280 print"     spazio per il file da scrivere.
290 print"iii) Usa lo schermo per rielaborare le
300 print"     informazioni via via acquisite.
310 print"     se si preme lo STOP, poi si deve
320 print"     RICOMINCIARE DA CAPO  !!!":print
330 dim frq(512),fi(511,1),pa(511),co$(255),h(513),l(255)
340 show$=" # device":rispsta$="8":gosub1550:dn=val(ri$)
350 print:print"inserisci il disco voluto e premi *"
360 geta$:ifa$<>"*"then360
370 open15,dn,15,"i":s=st>0
380 input#15,e,e$:s=sor(st<>64):ifeorsthen1510
390 gosub1290:rem apre file in lettura
400 gosub460:close15:end
410 :
```

```
420 rem"  ┌──────────────┐
430 rem"  | Compressione |
440 rem"  └──────────────┘
450 :
460 gosub1770:sysbs:close8:rem prima lettura file; statistiche in ling. macc.
470 :
480 rem statistiche e vettore per heap
490 :
500 lo=bs+96:hi=lo+256:min=peek(bs):max=peek(bs+1):nt=0
510 forx=mitoma:fr(x)=peek(x+lo)+256*peek(x+hi):iffr(x)thennt=nt+1:h(nt)=x
520 next:print"SLettura  file : OK":ifnt>1then540
530 print"un solo carattere ripetuto"h(1)"volte; cambia metodo !":goto910
540 print"ci sono"nt"caratteri diversi, con
550 print"codici ASCII variabili tra"mi"e"ma".
560 forx=nt+1to513:h(x)=512:next:fr(512)=9e9
570 :
580 rem creazione heap
590 :
600 print"Creo uno heap : ";:fork=int(nt/2)to1step-1:gosub730:print".";
610 next:print" OK":print"Creo l'albero : ";
620 :
630 rem creazione huf
640 :
650 root=ma+nt-1:forx=ma+1toro:print".";:k=3:iffr(h(3))>fr(h(2))thenk=2
660 a=fr(h(k)):fi(x,1)=h(k):pa(h(k))=x:h(k)=512
670 gosub730:k=1:fr(x)=a+fr(h(1)):fi(x,0)=h(1):pa(h(1))=x:h(1)=x
680 gosub730:next:lunghezza=fr(ro):print" OK"
690 print"Il file e' lungo "lu"byte":goto790
700 :
710 rem heapsort:ripristina heap che parte da k
720 :
730 c=h(k):b=fr(c):h=k:j=h+h:fory=1to1e9:iffr(h(j))>fr(h(j+1))thenj=j+1
740 ifb>fr(h(j))thenh(h)=h(j):h=j:j=h+h:next
750 h(h)=c:return
760 :
770 rem creazione codici
780 :
790 print"Creo i codici : ";:forx=mitoma:ifpa(x)=0then830
800 print".";:a$="":z=x:fory=1to1e9:w=pa(z):a$=chr$(49+(fi(w,0)=z))+a$
810 z=w:ifw=rothenl(x)=y:y=9e9
820 next:t=t+fr(x)*l(x):co$(x)=a$
830 next:al=nt*10-2:print" OK"
840 print:print"Il file compresso richiede"t"bit,
850 print"piu' altri"al"per l'albero.":t=int((t+al+7)/8)
860 ll=t+len(str$(lu)):gu=1-ll/lu:gu=int(100*gu)
870 print"In tutto usero'"ll"byte.":print"Rispetto ai"lu"originari c'e'"
880 print"un guadagno percentuale del"gu"%"
890 sh$="Creo il file compresso":ri$="si":ifgu<0thenri$="no"
900 gosub1550:ifri$<"s"thenreturn
910 print"nome del file su cui scrivere":gosub1410:print#9,lu
920 :
930 rem crea file compresso
940 :
950 open8,8,8,f1$:rem riapre il file in lettura
960 gosub1770:rem" 2 routine in L.M. su schermo
970 is=bs+360:rem sys bs compatta 256 bit partendo da is --> 32 byte al disco
980 giu$="Sqqqqqqqqq":rem scrivero' a partire da is, cioe' dalla decima riga
990 rem poke 253,n:sys bs+4 compatta solo 8*n bit e invia solo n byte al disco
1000 rem sys bs+35 legge dal disco 256 (o meno se st=64) e li poka da bs+64
1010 :
1020 rem scrittura albero
```

```
1030 :
1040 liv=0:scritt=is:pa=root:printgiu$;:gosub1080:goto1170
1050 :
1060 sc=sc+1:ifpa<=mathenprint"1";:goto1100
1070 print"0";
1080 h(li)=pa:li=li+1:pa=fi(pa,0):gosub1060
1090 pa=fi(h(li-1),1):gosub1060:li=li-1:return
1100 sc=sc+8:pa=pa/256:fory=1to8:pa=pa+pa:printchr$(pa+48);:pa=pa-int(pa):next
1110 ifsc<is+257thenreturn
1120 sysbs:sc=sc-256:printgiu$;:forz=istosc-1:pokez,peek(z+256):next
1130 printspc(sc-is);:return
1140 :
1150 rem scrittura testo
1160 :
1170 il=bs+64:ul=il+255:rem inizio dati letti da disco, ultimo dato gia' letto
1180 forx=1to1e9:lu=lu-256:sysbs+35
1190 iflu<=0thenx=9e9:ul=ul+lu
1200 fory=iltoul:z=peek(y):printco$(z);:sc=sc+l(z)
1210 ifsc<is+257then1240
1220 sysbs:sc=sc-256:printgiu$;:forz=istosc-1:pokez,peek(z+256):next
1230 printspc(sc-is);
1240 next:next:poke253,(sc-is+7)/8:sysbs+4
1250 close8:close9:print"█Fatto":return
1260 :
1270 rem" chiede file, cerca tipo; su C16,Plus4,C128 aggiungere: directory
1280 :
1290 print"Dammi il nome del file da leggere"
1300 inputfl$:l=len(fl$):ifl=0then1290
1310 ty$=",s":gosub1360
1320 ife=64thenclose8:ty$=",p":gosub1360
1330 ife=64thenclose8:ty$=",u":gosub1360
1340 ifethen1510:rem 64 - file type mismatch
1350 return
1360 open8,dn,8,fl$+ty$+",r":input#15,e,e$:return
1370 :
1380 rem chiede un nome per il file su cui scrivere
1390 :
1400 print"Dammi un altro nome per il file
1410 sh$="(senza virgole, *, ?, e simili)":ri$="":gosub1550:fs$=ri$
1420 nome$=fs$+ty$+",w":open9,dn,9,nome$:input#15,e,e$
1430 ife=0thenreturn
1440 close9:if(e<>63)then1510:rem 63 significa file exists
1450 print"Sul disco c'e' gia' un file col nome █"fs$
1460 sh$="lo sopprimo":ri$="no":gosub1550:ifri$<"s"then1400
1470 a$="s"+d$+":"+fs$:print#15,a$:goto1420
1480 :
1490 rem problemi col disco
1500 :
1510 print"probl. col disco : "e$:close15:clr:end
1520 :
1530 rem input guidato
1540 :
1550 printspc(len(sh$)+4)ri$"█":print" █"sh$;
1560 input"█ ";ri$:return
1570 :
1580 data0:rem offset=0; sys bs da' fr(car.): low da $460, hi da $560
1590 :
1600 data169,0,170,157,96,4*,157,96,5*,232,208,247,162,8,32,198
1610 data255,32,165,255,170,254,96,4*,208,3,254,96,5*,165,144,201
1620 data64,208,238,32,204,255,162,255,232,189,96,4*,29,96,5*,240
1630 data247,142,0,4*,162,0,202,189,96,4*,29,96,5*,240,247,142
```

```
1640 data1,4*,96
1650 data-1
1660 :
1670 data0:rem offset=0; sys bs compatta 256 bit da bs+$168 e li scrive
1680 rem poke 253,n e sys bs+4 ne compatta 8*n da bs+$168 e li scrive
1690 rem sys bs+35 legge 256 (o meno se st=64) e li pone da bs+$36
1700 :
1710 data162,32,134,253,162,9,32,201,255,162,0,169,1,188,104,5*
1720 data192,49,42,232,144,247,32,168,255,198,253,208,238,76,204,255
1730 data234,234,234,162,8,32,198,255,162,0,160,0,32,165,255,157
1740 data64,4*,232,165,144,201,64,240,3,200,208,240,76,204,255,234
1750 data-1
1760 :
1770 print"S":reada:sy=bs+a:forx=sytole9:reada$:ifa$="-1"thenreturn
1780 a=val(a$):ifright$(a$,1)="*"thena=a+(bs-1024)/256
1790 pokex,a:next
1800 end

ready.

1 data200->42,100->18,50->8,25->3; modo fast, seme 5
100 printchr$(147)"demo di un programma di
110 print"h e a p - s o r t":print
120 input"quanti elementi";n:input"seme random";s
130 print"preparo gli elementi
140 dimb%(n),a$(n):r=rnd(1+abs(s))
150 input"vuoi vederli";a$:r=a$>"no
160 forx=1ton:a$=""
170 fory=2to4+5*rnd(1):a$=a$+chr$(65+26*rnd(1)):next
180 ifrthenprinta$"*";
190 a$(x)=a$:next:r=fre(0):print:print"fatto; premi un tasto ...
200 get a$:if a$="" then 200
210 print"o.k.; sono partito
220 ti$="000000":gosub50000:printint(ti/60+.5)"secondi"
230 print"per riordinare"n"elementi.
240 input"vuoi vederli";a$
250 ifa$>"no"thenforx=1ton:printa$(b%(x)):next
260 end
270 :
49960 :
49970 rem heap sort su a$(1), ..., a$(n)
49980 cnfrnti nelle righe:on a goto 50080,50090
49990 :
50000 forx=1ton:b%(x)=x:next:ifn<2thenreturn
50010 m=n:forx=int(n/2)to1step-1:gosub51000:next
50020 ifn=2then50050
50030 form=n-1to2step-1:x=b%(m+1):b%(m+1)=b%(1):b%(1)=x
50040 x=1:gosub51000:next
50050 x=b%(2):b%(2)=b%(1):b%(1)=x:return
50060 :
50070 rem la subroutine seguente ripristina lo heap
50080 rem sul sottoalbero che nasce dal nodo #k
50090 rem e arriva al massimo al nodo #m.
50100 rem lo heap e' alla rovescia ....
50110 :
51000 z=b%(x):h=x:j=h+h:fory=1tole9
51010 ifj<mthenifa$(b%(j))<a$(b%(j+1))thenj=j+1
51020 ifa$(z)<a$(b%(j))thenb%(h)=b%(j):h=j:j=h+h:ifj<=mthennext
51030 b%(h)=z:return

ready.
```

VERY IMPORTANT PROGRAM PER C/64

Da anni il Vip Terminal rappresenta il programma di comunicazione più diffuso ed utilizzato per C/64 e C/128 in modo 64. Vediamone insieme le caratteristiche fondamentali

di Marco Miotti

Supponiamo che il signor Matteo Motta di Milano sia utente di BBSX, sempre di Milano (tale BBS non esiste, come il signor Motta, per cui non provate a cercarla) che si trova in collegamento con altre BBS di Roma, Napoli e Palermo.

Il signor Motta ha un grosso problema: non sa come fare per utilizzare contemporaneamente il cavo parallelo per lo speed-dos e l'interfaccia seriale sul suo C/64. Colpito da lampo di genio chiama BBSX e lascia un messaggio con richiesta d'aiuto relativa al suo problema; il giorno dopo tale messaggio sarà giunto anche a Roma, Napoli e Palermo alle altre BBS collegate a BBSX e il giorno dopo ancora le consociate delle altre città riceveranno da Roma, Napoli e Palermo lo stesso messaggio. In capo ad un paio di giorni almeno qualche decina di migliaia di persone potranno leggere la richiesta d'aiuto di Matteo Motta.

Ovviamente l'eventuale risposta tornerà, seguendo il percorso inverso, a BBSX, dove il signor Motta la potrà leggere e valutare.

Tale modo di procedere si chiama **Echomail** ed è un servizio molto diffuso in tutte le BBS **del mondo**, il che consente anche il collegamento con i paesi esteri.

Commodore Computer Club ed in particolare la *Systems Editoriale* ha presentato nei numeri scorsi un'iniziativa che fornisce al lettore un modo moderno di interpellare la redazione. E' infatti aperta, sia pure ad uno stadio puramente sperimentale, Bbsystems, la BBS marcata Commodore Computer Club e Personal Computer.

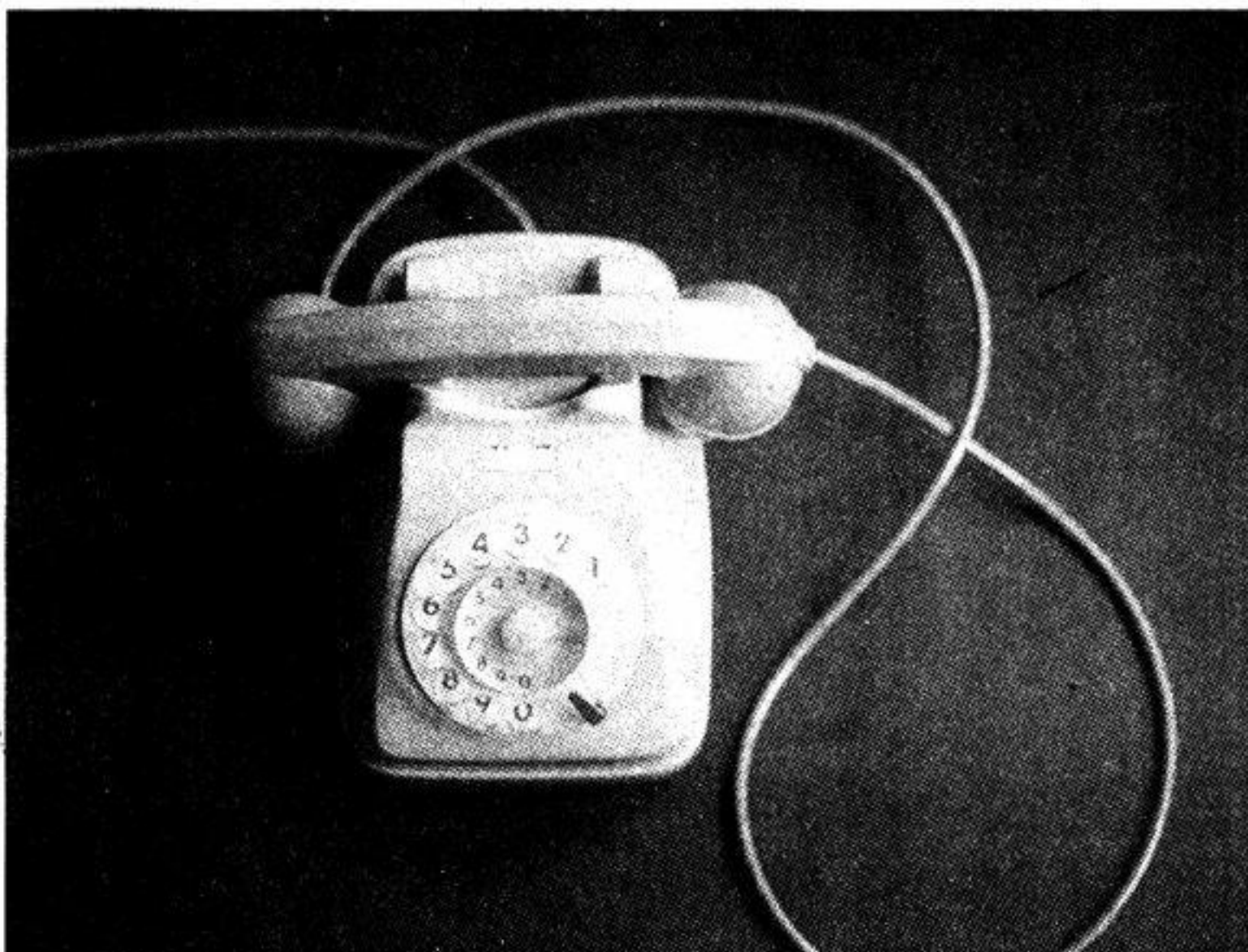
Grazie al nuovo servizio (per la descrizione del quale rimandiamo il lettore ad altri articoli, passati e futuri) il lettore munito di modem ad *almeno* 300 Baud potrà collegarsi e porre domande, leggere le risposte relative al suo quesito ed anche a quello degli altri lettori, dare suggerimenti relativi alle riviste di casa Systems, prelevare e/o mandare file per C/64-128, Amiga e Ms-Dos nonché utilizzare quello che rappresenta il servizio più utile e divertente di una BBS: lo scambio dei messaggi fra utenti.

Il C/64 e il telefono

Lo scopo dell'articolo non è, però, la descrizione di una BBS o le modalità di collegamento con un modem; la necessaria introduzione serviva per arrivare al tema vero e proprio della nostra chiacchierata: il C/64; come collegarlo al mondo esterno?

Poveri noi; il computer da casa più diffuso del mondo (almeno fino ad ora) presenta numerosi problemi per gestire degnamente le comunicazione.

All'inizio degli anni '80, quando la Commodore Business Machine annun-

ciò il prodigioso Commodore 64 (ricordo che in redazione arrivò il prototipo n. 15 che aveva lo stesso colore del Vic 20 ed era di quattro centimetri più alto rispetto ad un 64 standard vecchio modello) la gestione dei dati era ancora affidata a tiratissimi processori a 8 bit, tra i quali si trovava anche il 6502.

Premesso che già alla fine degli anni '70 le risorse tecnologiche tendevano ad orientarsi verso la realizzazione di macchine con un parallelismo di almeno 16 bit, il mondo dell'home computer doveva per forza adattarsi al livello medio allora più diffuso.

Il C/64 veniva quindi equipaggiato con un prodigioso **6510** a 8 bit e funzionante a "ben" un megahertz, quando già lo Z80 poteva quasi quadruplicare la velocità di elaborazione.

Nel 1990 le cose sono un po' cambiate. Un computer Ms-Dos compatibile equipaggiato con 80386 (32 bit) e funzionante a 33 Mhz costa quanto sei Commodore 64 del 1983 e rappresenta una spesa che, per validi motivi professionali, può essere se non direttamente affrontata almeno considerata e profondamente valutata. Il C/64 ed il fratello maggiore C/128 risentono, ahimè, pesantemente delle limitazioni imposte dal livello tecnologico implementato che, a distanza di dieci anni, è davvero da considerarsi nè superato nè obsoleto ma addirittura preistorico.

Ed ecco che, puntuali, cominciano a fischiarci le orecchie, ma i 64isti non devono prendere a male tali considerazioni; chi ha acquistato un C/64 nel lontano 1984 è costretto ad ammettere che gli anni passano, i figli crescono, le mamme imbiancano e i computer invecchiano. La logica conseguenza è che in un mercato così all'avanguardia come quello dell'informatica, restare indietro di anche solo qualche mese comporta notevoli penalità nell'acquisizione dei termini tecnologici che, nel frattempo, si sono presentati.

Il discorso relativo alle BBS contribuisce a girare (dolorosamente) il coltello nella piaga; chi è costretto ad operare con un modem da 300 Baud spesso impreca contro la Sip ed i progettisti del CIA 6526, il chip che, nel C/64, gestisce le operazioni di I/O.

Limiti di velocità

La massima velocità di comunicazione prevista per la porta utente del C/64 è di soli 2400 bps che, però, noi siamo riusciti a far funzionare **solo** in uscita e con collegamento **diretto** al computer destinazione. Abbiamo infatti sviluppato un sistema Hard/Soft per la conversione, in formato Ms-Dos, degli articoli scritti con un C/64.

In seguito alla procedura, gli stessi articoli possono essere direttamente impaginati a video (prodigi della tecnologia).

Esiste, invece, qualche problema inviando i dati da PC a C/64; la massima velocità alla quale il piccolo Commodore può ricevere e contemporaneamente gestire efficacemente il drive è di 1200 Baud che, tutto sommato, non è neanche tanto male.

Il discorso cambia totalmente se ci si affida all'**adattatore telematico 6499** della Commodore.

Tale dispositivo è stato infatti progettato **solo** per il funzionamento a **300** Baud in collegamento standard oppure a 1200/75 bps per il collegamento con Videotel (la velocità di 75 Baud in uscita consente all'elaboratore Videotel di gestire contemporaneamente numerosi utenti...).

Qualsiasi collegamento con il 6499, quindi, risulta penalizzato da una serie di problemi di linee telefoniche e, soprattutto, di mancanza di file transfer.

Vip terminal

Da qualche anno "circola" nel mondo del C/64 il **Vip terminal** che altri non è se non il **miglior programma** di comunicazione per C/64 mai visto (e se qualcuno osa sostenere il contrario... lo faccia; è sempre ben accetta qualsiasi critica costruttiva). Progettato (forse) intorno al 1986, il programma faceva parte di un pacchetto contenente una serie di prodotti di tipo **Desktop**, che significa "scrivania"; il Vip Terminal è comunque il programma che, nel pacchetto Vip, ha riscosso maggior successo in quanto consente all'utente la totale interattività con tutti i modelli di modem presenti allora per C/64 tanto che tutti i modem costruiti in seguito, **escluso** l'adattatore telematico 6499 che costituisce un prodotto a sè stante, furono progettati per funzionare con Vip Terminal che cominciò a girare anche in versione "freezata", cioè non più assieme agli altri prodotti del pacchetto.

Inoltre supporta anche i modem di tipo **Hayes** compatibili, dispositivi che prevedono l'utilizzo di una interfaccia seriale standard RS-232 per il collegamento al modem stesso e che ne fanno virtual-

Modem e BBs

Finalmente la comunicazione via modem raggiunge il livello di fenomeno di massa, grazie ad una tangibile riduzione dei prezzi hardware e ad una contemporanea diffusione del concetto di "conversazione elettronica".

Da qualche tempo, infatti, la soglia dei **300 Baud** è stata abbattuta e ora, grazie proprio a questo fenomeno, un modem da **1200 Baud** costa poco più che un paio di giorni in albergo a Rimini in pieno Agosto (con il vantaggio, in più, di non sentire caldo). Oggigiorno 1200 bps cominciano ad essere già stretti per l'utente medio che guarda già verso gli orizzonti del **2400** e 4800.

Logicamente il discorso non poteva restare al livello di puro e arido scambio di programmi tra (pochi) smanettoni, ed è così che sono nate le **BBS** (Bulletin Board System), un servizio speciale destinato a coloro che intendono usare il modem come mezzo di comunicazione con il resto del mondo.

Grazie a particolari accordi tra i gestori delle BBS, il discorso si allarga talvolta a livello nazionale e lo spirito di cooperazione presente in tutti i **SysOp** consente una rapida diffusione delle informazioni inviate.

mente il dispositivo universale per eccellenza dedicato alla comunicazione.

Un completo sistema di menu consente all'utente di muoversi attraverso le opzioni del programma in modo da settare i più svariati parametri di trasmissione. I menu si ottengono semplicemente utilizzando i tasti funzione.

F1 - Attiva l'help del programma dove, una volta inserito il **disco sistema** di Vip Terminal, è possibile ottenere informazioni sugli altri menu ottenibili, così da capire all'istante quali sono le opzioni da settare per il funzionamento corretto.

F2 - Tramite questa opzione si accede alla configurazione vera e propria di Vip Terminal, divisa in due sottomenu: il primo prevede i valori che influiscono sull'aspetto grafico del vip terminal, compreso un ottimo emulatore di 64/80/106 colonne per consentire l'ottimale utilizzo di elaboratori "grossi" che funzionano, appunto, con quella risoluzione.

E' prevista inoltre la possibilità di definire colori, battute udibili sulla tastiera, icone per i vari menu a fondo schermo, il word wrap ed altro...

Il secondo sottomenu serve per impostare i parametri di trasmissione, partendo dal baud rate, parità ecc. fino alla gestione del protocollo di handshaking e alla velocità di trasmissione "fast".

F3 - E' l'opzione destinata all'emulazione terminale vera e propria. Si accede infatti alla modalità **talk** dove tutto ciò che si digita automaticamente viene inviato alla porta utente, verso il modem. Anche qui esistono diversi comandi da impartire e possono essere visualizzati in una breve schermata di help semplicemente premendo la sequenza di tasti *Ctrl* e *h*. Questa è la modalità da attivare ogni volta che si intenderà colloquiare. L'**autodial**, previsto ad esempio per tutti i modem Hayes compatibili, posizionerà automaticamente il Vip Terminal in modalità **talk** una volta stabilita la presenza della portante, quando cioè il collegamento è avvenuto.

F4 - Viene utilizzato per la regolazione di data, ora e sveglia (per ricordarsi quando la Sip diventa troppo cara...).

F5 - Tramite questa opzione si accede ad un menu orientato alla gestione della stampa di file ASCII. E' prevista l'installazione di tre diversi tipi di stampante: la 1525 (compatibile Mps 801 e 803), ASCII e CBM; nonché l'indirizzo primario e secondario.

F6 - E' l'opzione per la definizione delle macro. Queste sono semplicemente delle stringhe di caratteri che l'utente si stanca di inviare ogni volta alle bbs e che il computer provvede a gestire. Un classico esempio di tale applicazione è l'inserimento del nominativo e della password utente.

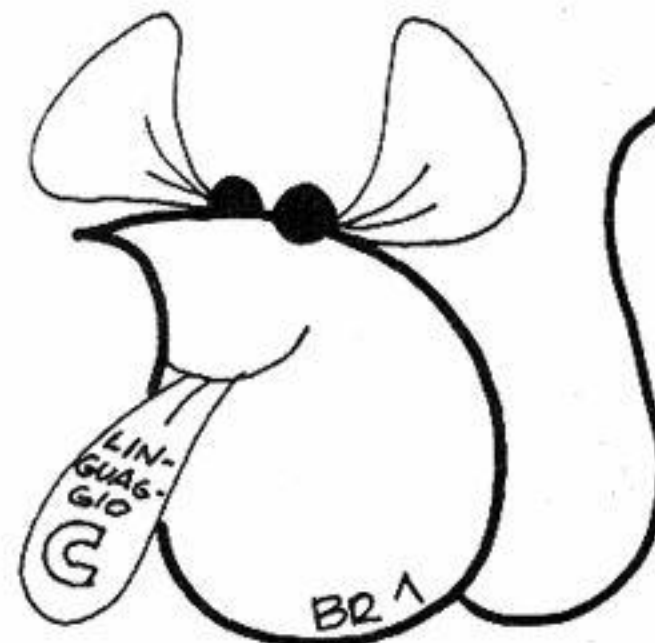
F7 - Mediante quest'opzione è possibile accedere ad un comodo menu di gestione dischi, grazie al quale si può formattare, copiare, rinominare ecc.

Sempre all'interno del menu troviamo la possibilità di salvare ed eventualmente ricaricare l'ambiente operativo (comprese le macro ed i numeri di telefono) utilizzati attualmente.

E' definita anche un'area detta **workspace** che può essere aperta in modalità **talk** così da memorizzare tutto ciò che accade dal momento in cui ci colleghiamo in poi.

Sempre in questo menu esiste l'importantissima possibilità di **convertire** i dati in arrivo in formato CBM; come tutti infatti saprete l'ASCII Commodore differisce da quello standard; tale opzione consente quindi di operare le trasformazioni del caso, affinché tutto funzioni per il meglio.

F8 - Infine è presente il database per la gestione dei numeri di telefono da chiamare. Ve ne sono ben 16 e ciascuno di questi contiene informazioni relative al nominativo, al numero di telefono, baud rate, parità e perfino password da assegnare in automatico.

E' possibile definire il modello del modem che si sta utilizzando, così da rendere il tutto molto flessibile e pratico.

La sfera di cristallo

Perché è opportuno usare Vip Terminal? E' molto semplice: tale programma consente una rapida ed efficace gestione del dispositivo di comunicazione, assieme ad un semplice setup di ambiente operativo. Ma soprattutto è l'unico che prevede ed utilizza molto bene (anche se lentamente) il protocollo **Xmodem** che consente l'implementazione di file transfer anche molto lunghi (sempre con sufficiente tempo di collegamento) e poi, ovviamente può funzionare bene anche a 1200 Baud.

E' ovvio che Vip Terminal rappresenti la soluzione ideale per gli utenti che possiedono un modem dedicato al C/64, (purchè non sia l'adattatore telematico 6499) e che vogliono quanto meno provare ad utilizzarlo con le BBS della zona; è chiaro che più di tanto il C/64 (poverino) ed il C/128 non possono fare in quanto l'hardware è quello che è; la tecnologia non aspetta tempo e, tra un anno al massimo, il "gap" che separa il C/64 dai suoi cugini maggiori (parliamo di Amiga e di PC compatibili) diventerà eccessivo.

E' vero che il primo amore non si scorda mai (e che il C/64 resta un valido strumento hobbistico) ma se non volete perdere l'appuntamento con il futuro cercate di valutare la possibilità di utilizzare sistemi dotati di caratteristiche superiori.

Se, quindi, possedete il C/64 e volete effettuare qualche piccolo esperimento nel mondo della telematica, procuratevi un buon modem (che potrete, comunque, usare in seguito su altri computer), l'interfaccia Rs-232 ed il Vip Terminal.

LA POSTA DEL C/128

(a cura di Domenico Pavone)

Che notazione e'?

Nella rubrica di Posta 128 sul numero 74 della rivista, si accenna ad una notazione BCD (Binario Codificato Decimale) indispensabile per disporre di un orologio affidabile, non legato alla variabile TI$ del basic. Potreste essere più precisi? A che cosa serve, e come si adopera sul C/128?
(R. B. - Mestre)

La notazione **BCD**, a dispetto delle apparenze, rappresenta una convenzione numerica di facile gestione, purchè si abbia un minimo di abitudine a manipolare bit, byte e "roba" del genere.

Il modo più semplice per capire di che cosa si tratti, è ricorrere ad un esempio concreto.

Consideriamo un numero espresso in notazione binaria: **0101 , 0111**.

Come dovrebbe essere noto, ci troviamo di fronte ad un byte, in quanto composto da 8 bit.

Lo spazio interposto tra i primi quattro bit ed i restanti quattro, scompone il numero in due **Nibble**, entità che per l'appunto rappresenta un raggruppamento di 4 bit.

Il nibble cosiddetto **alto** (o anche "più significativo") è quello posto a sinistra, l'altro viene comunemente definito come "nibble **basso**" (o "meno significativo"). Il byte prima visto, tradotto in notazione decimale, corrisponde ad **87**. Si pensi, ora, alla procedura più semplice per trasformarlo in esadecimale: basta tradurre, in quella notazione, i singoli nibble, e si otterrà quanto desiderato.

In pratica, ricordando che con il simbolo di percentuale (**%**) viene indicata la notazione binaria e con il dollaro (**$**) quella esadecimale, avremo...

%0101 = $5 (nibble alto)
%0111 = $7 (nibble basso)

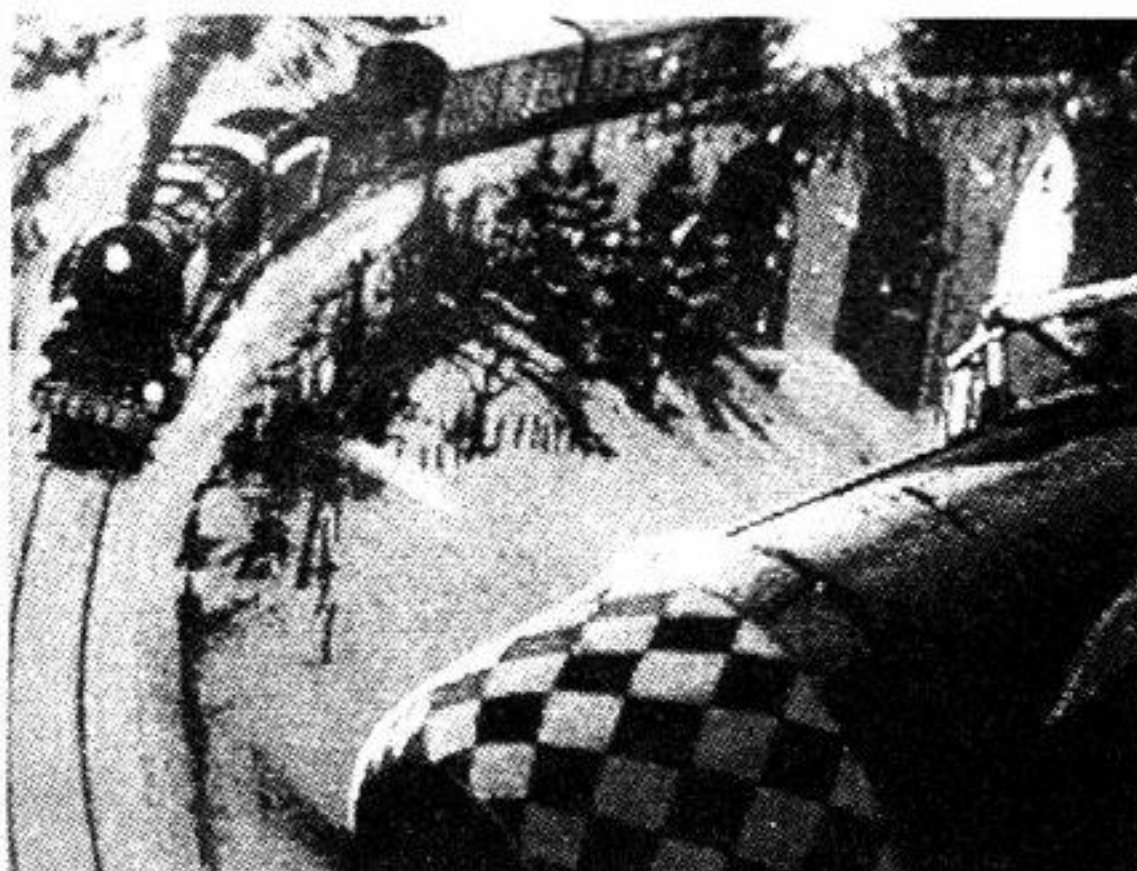
... e, in definitiva, **$57** esadecimale.

Con la notazione **B**inaria **C**odificata **D**ecimale (**BCD**), si assume, per convenzione, che ogni nibble possa esprimere un valore decimale compreso tra **0** e **9**, che rappresenta quindi una singola cifra in questa notazione.

L'esempio prima visto a proposito della traduzione in esadecimale di un byte, può dunque essere applicato senza alcuna modifica al Bcd, in quanto il nibble alto varrà **5**, quello basso **7**, e la cifre risultante sarà proprio 57. Non si dimentichi, però, che si tratta pur sempre di una "*convenzione* ", per cui se è vero che %01010111 può essere "inteso" come un valore Bcd 57, il suo reale contenuto varrà sempre 87, così come $57 (esadecimale) non sarà uguale a 57 decimale.

Un byte, inteso in Bcd, può in definitiva assumere un valore compreso tra **0** e **99** (e non più tra **0** e **255**).

Una tipica applicazione pratica di quanto descritto, è proprio la gestione del cosiddetto **TOD** (**T**ime **O**f **D**ay), il preciso orologio interno in dotazione al C/128 (ed al C/64).

Anzi, per la precisione, di orologi ce ne sono ben due, i cui registri si trovano locati agli indirizzi **$DC08 - $DC0B**

e **$DD08 - $DD0B** di banco 15. In queste locazioni, opportunamente "attivate", vengono immagazzinati ore, minuti, secondi e decimi di secondo proprio in accordo alla notazione Bcd.

Maggiori dettagli sui Clock del CIA possono essere desunti da un articolo pubblicato proprio sullo stesso numero 74, anche se l'applicazione proposta in quella sede è riferita al C/64.

In queste pagine, comunque, è riportato un breve disassemblato che, opportunamente copiato adoperando il Monitor per linguaggio macchina incorporato nel computer (comando **A 1300**....), consente la visualizzazione continua (sganciata dal basic grazie alla strasolita manipolazione dell'**Interrupt**) del clock in reverse sullo schermo anche sul C/128 in modo 128. Per attivare la routine, basta impartire **Sys 4864** da banco 15, mentre **Run / Stop + Restore** provoca la sua disattivazione.

L'ora impostata (le 8 e 15 antimeridiane) può essere modificata, oltre che intervenendo sul disassemblato, Pokando direttamente da basic ore e minuti nelle locazioni **4865** e **4870**.

Naturalmente, in accordo con quanto prima espresso a proposito del Bcd.

Per "studiare" la procedura seguita, è sufficiente riferirsi ai commenti posti a fianco delle singole istruzioni, integrando poi il tutto con quanto esposto dall'articolo sul n. 74.

Si presti attenzione al fatto che, per tradurre nei corrispondenti codici di schermo i byte Bcd, viene adoperato un "offset" **$B0** (dec. **176**), in modo da ottenere i caratteri in reverse.

Piu' ram, meno problemi?

Mi sono procurato una espansione di memoria (modello 1750) per il mio C/128. Potreste dirmi se esistono in memoria delle Rom routines che la sfruttino al meglio?
(Qualcuno)

Nelle Rom del 128, originariamente in previsione di fantastici futuri sviluppi (...), è presente un piccolo raggruppamento di 11 registri, dedicati alla gestione del cosiddetto **Chip REC** (**R**am **E**xpansion **C**ontroller).

Chip che però... non c'è.

Nel senso che non è proprio installato nel computer, anche se esistono dei comandi basic (**Stash, Fetch e Swap**) ed una routine del kernal (**$F7A5**) che svolgono la loro azione manipolando i suddetti registri.

Tutto però va a posto quando si adopera una espansione come la **1700** (128 Kbyte) oppure la **1750** (512 Kbyte), in quanto il chip fa parte del loro corredo hardware.

La Ram supplementare, comunque, non è accessibile direttamente al processore del 128: non si pensi di trovarsi con (per esempio) un bank 0 di 512 Kappa!

L'unica "porta" attraverso la quale accedere alla Ram dell'espansione, è rappresentata dal REC ed i suoi registri. Il che, ad onor del vero, non è poi poco.

Si può pensare a questo chip come ad un agente superspecializzato nello spostamento di dati.

Le uniche quattro operazioni possibili, infatti, sono le seguenti:

* Trasferimento di dati dalla Ram del computer a quella dell'espansione **(Stash)**
* Trasferimento dalla Ram dell'espansione a quella interna (**Fetch**).

* Scambio contemporaneo di dati tra un'area di memoria del 128 ed una dell'espansione (**Swap**).
* Comparazione tra un'area di memoria del computer ed una dell'espansione (**Verify**). Le prime tre possono essere attivate anche da Basic con i già accennati comandi, mentre Verify è implementabile solo da linguaggio macchina (si veda descrizione dei registri).

Come si sarà notato, le performance del Rec corrispondono in pratica a quelle comunemente offerte dai disk drive, motivo per cui è più che corretto riferirsi al modulo di espansione intendendolo come Ram Disk.

Il **CP/M** (pace all'anima sua) addirittura, è in grado di gestire espressamente questa unità come drive virtuale, invocandola come **drive M:**.

Per sfruttarla in modo 128, l'unica via è invece il solito... far da sè.

Caratteristica peculiare di questa **Ram Disk** (ormai pos-

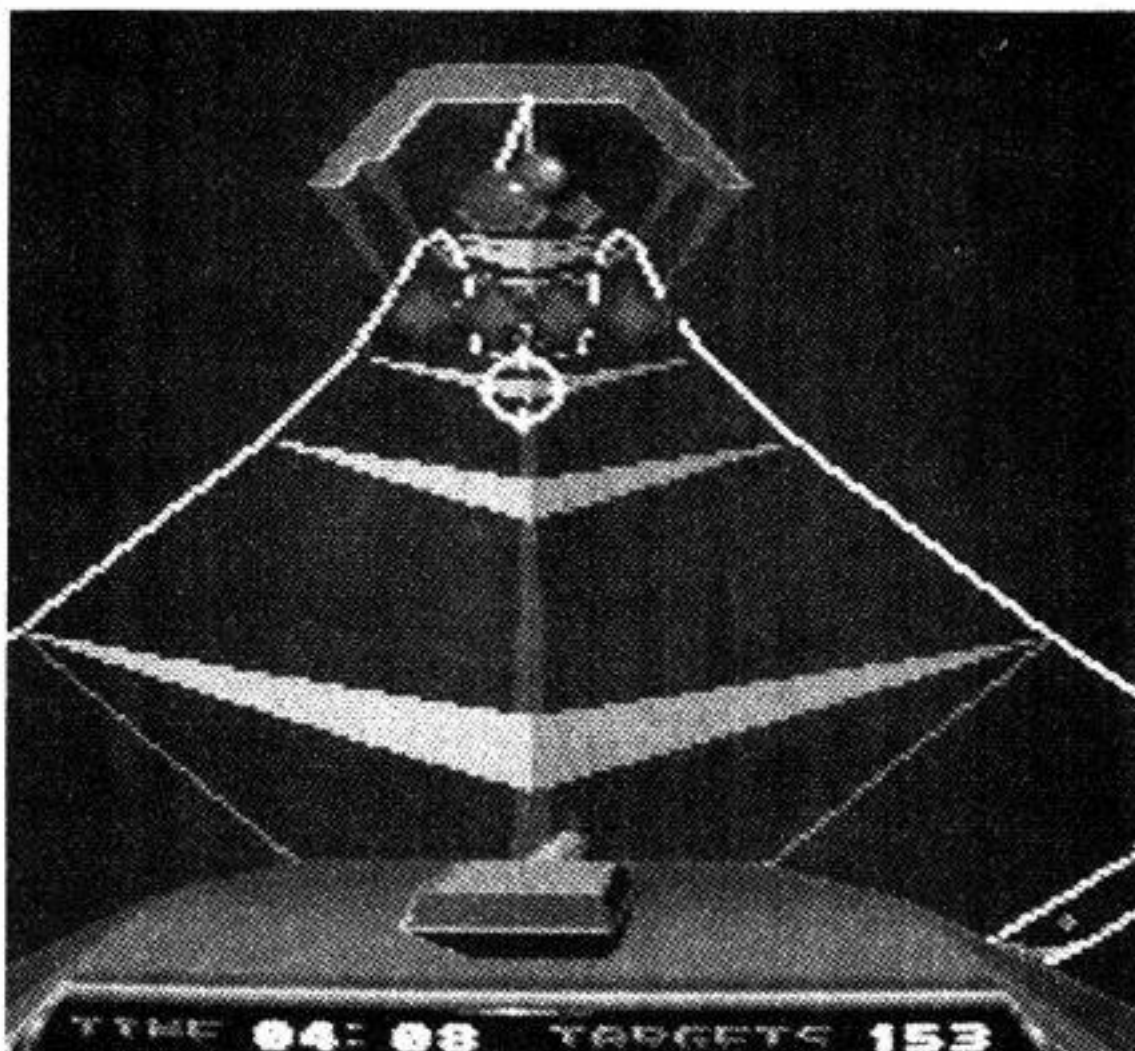



TAVOLA DEI REGISTRI DEL CHIP REC

$DF00 (dec. 57088)

Detto **Registro di Stato**, a sola lettura.

* Il bit **5** (non si dimentichi che i bit vanno da **0** a **7** e **non** da 1 a 8) risulta settato (posto ad 1) se è stato riscontrata una diseguaglianza durante un'operazione di verifica. Il bit è riazzerato dopo ogni lettura del registro.

* Il Bit **6** indica, quando risulta settato, che è stata completata un'operazione di trasferimento o verifica.

* Il bit **7** segnala invece che il REC ha generato un interrupt, cosa che avviene tanto nel caso di un errore di verifica, quanto dopo la conclusione di un trasferimento di dati (vedi anche $DF09).

$DF01

Detto **Registro Comandi**.

* I bit **1** e **0** determinano il tipo di operazione che il Rec deve svolgere (00 = Stash, 01 = Fetch, 10 = Swap, 11 = Verify).

* Il bit **4** può essere settato, nel qual caso l'operazione specificata dai bit 0 e 1 ha inizio immediatamente dopo che si è settato il bit **7**. Se viene azzerato, l'esecuzione è posticipata, nel senso che ha inizio solo dopo un'intervento in scrittura nel registro di controllo **$FF00** (MMU), che modifica la configurazione di banco del 128. Solo per applicazioni particolari.

* Il bit **5** controlla lo stato finale dei registri (vedi più avanti) contenenti gli indirizzi di base (che vengono incrementati automaticamente dal Rec) ed il conteggio dei byte coinvolti nel trasferimento (che viene invece decrementato).

Se questo bit è azzerato (lo è per default), dopo la conclusione delle operazioni i suddetti registri manterranno la loro condizione finale, in caso contrario (bit settato) verranno ripristinati i valori in essi contenuti prima del trasferimento. Utile nel caso occorra ripetere più volte un'operazione riguardante la stessa area di memoria.

* Il bit **7** avvia effettivamente l'operazione nel momento in cui viene settato, per cui va impostato solo dopo aver inizializzato tutti i registri necessari.

$DF02 - $DF03

Questi registri precisano l'indirizzo di inizio, nella memoria del C/128, per la corrente operazione del Rec.

L'indirizzo, come di consueto, deve essere nel formato byte basso (in **$DF02**) byte alto (in **$DF03**), e varrà la configurazione di banco al momento attiva.

$DF06

I primi tre bit di questo registro indicano a quale banco (di 64K) della memoria di espansione fa riferimento l'indirizzo contenuto nei due precedenti registri. Come ovvio, l'espansione 1700 avrà solo 2 banchi, contro gli otto della 1750.

Le possibili combinazioni dei bit sono:

BIT	BANCO
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

$DF04 - $DF05

Devono contenere l'indirizzo di inizio nel modulo di espansione per la corrente operazione del Rec, anche qui in formato basso / alto.

Si badi che la memoria nell'espansione è suddivisa in "banchi" di 64K ciascuno (vedi registro seguente), per cui se durante un'operazione l'indirizzo (a mano a mano che viene incrementato) supera $FFFF, torna a zero, ma il numero di banco (dell'espansione) viene automaticamente incrementato, senza creare problemi di sorta.

$DF07 - $DF08

Byte basso e byte alto del valore indicante il numero di byte da trasferire (o verificare). Se l'operazione viene condotta a termine con successo, e sempre che non si sia intervenuto sul bit 5 del registro $DF01, queste locazioni conterranno **1** e **0**.

$D0F9

Registro di controllo degli **Interrupt**.

Il Rec genera un IRQ dell'8502 solo in due eventualità: quando viene riscontrata una diseguaglianza tra bytes in una operazione di Verify, nonchè quando viene completato un trasferimento di dati (o una verifica).

Settando il bit **5** di questo registro viene abilitata la possibilità di generare un interrupt per il primo caso, mentre settando il bit **6** si attiva la seconda possibilità. Ovviamente, possono essere settati entrambi, comunque l'attivazione non sarà esecutiva fino a che non viene settato anche il bit **7**, che può essere visto come un "interruttore" generale per gli interrupt.

$DF0A

I bit **6** e **7** di quest'ultimo registro, controllano l'incremento automatico degli indirizzi di base durante i trasferimenti oppure verifiche. Le impostazioni possibili sono:

00 = Vengono incrementati tanto gli indirizzi di memoria di sistema che quelli riguardanti l'espansione (condizione di default).

01 = Incremento del solo indirizzo di memoria del C/128.

10 = Incremento del solo indirizzo di memoria dell'espansione.

11 = Non vengono incrementati entrambi gli indirizzi di base.

Adoperando il secondo o terzo settaggio, si ha in pratica la possibilità di riempire con un valore prestabilito (ed a tempo di record) interi banchi di memoria di sistema o dell'espansione.

siamo chiamarla così), è l'estrema velocità dei trasferimenti, superiore anche a quella dello stesso processore centrale.

Tra l'altro, quando il REC svolge il suo lavoro, prende totalmente "in mano" il sistema, escludendo l'8502, ovvero operando nel cosiddetto modo **DMA** (**D**irect **M**emory **A**ccess), ben più noto agli utenti di Amiga.

Per la cronaca, questo tipo di accesso alla memoria consente al Rec una velocità "di crociera" di **1 milione di byte al secondo** per i trasferimenti (!), che si traduce in circa 1/16 di secondo per salvare in Ram Disk un intero banco del C/128 (64K).

Per l'utilizzo pratico in Basic, basta dare un'occhiata al manuale alle voci Stash, Fetch e Swap, mentre i più esperti possono aiutarsi con il riquadro - *pro - memoria* di queste pagine per elaborare delle routines LM che gestiscano al meglio le capacità del Rec.

Senza, però, dimenticare un particolare:

Se il C/128 è in pratica da annoverarsi nella categoria degli "ex", non sarà certo un modulo di espansione a farlo resuscitare...

```
100 REM LISTATO CLOCK LM PER C128
110 REM
120 :
130 BANK 15: FOR X = 0 TO 96: READ A
140 POKE X + 4864, A: K = K + A: NEXT
150 IF K = 8339 THEN SYS 4864: END
160 PRINT "ERRORE NEI DATA!"
170 :
180 DATA 169, 008, 141, 011, 220, 169, 021
190 DATA 141, 010, 220, 169, 000, 141, 009
200 DATA 220, 141, 008, 220, 120, 169, 031
210 DATA 162, 019, 141, 020, 003, 142, 021
220 DATA 003, 088, 096, 160, 186, 173, 011
230 DATA 220, 041, 127, 032, 083, 019, 141
240 DATA 000, 004, 142, 001, 004, 140, 002
250 DATA 004, 173, 010, 220, 032, 083, 019
260 DATA 141, 003, 004, 142, 004, 004, 140
270 DATA 005, 004, 173, 009, 220, 032, 083
280 DATA 019, 141, 006, 004, 142, 007, 004
290 DATA 173, 008, 220, 076, 101, 250, 072
300 DATA 041, 015, 009, 176, 170, 104, 074
310 DATA 074, 074, 074, 009, 176, 096
320 END
```

DISASSEMBLATO CLOCK PER C128

Predispone	**orario**	
1300	LDA #$08	Ore 8 AM...
1302	STA $DC0B ,	
1305	LDA #$15	Minuti 15...
1307	STA $DC0A	
130A	LDA #$00	Secondi 0...
130C	STA $DC09	
130F	STA $DC08	Start!
Dirotta	**Interrupt**	
1312	SEI	Inserisce
1313	LDA #$1F	indirizzo
1315	LDX #$13 $131F	(inizio
1317	STA $0314	nostra routine)
131A	STX $0315	nel vettore di
131D	CLI	interrupt....
131E	RTS	e Return.
Aggiorna	**visualizzazione**	
131F	LDY #$BA	Car.due punti.
1321	LDA $DC0B	Preleva ore.
1324	AND #$7F	Esclude bit 7.
1326	JSR $1353	Traduce in Ascii
1329	STA $0400	e stampa su
132C	STX $0401	schermo.
132F	STY $0402	Separatore.
1332	LDA $DC0A	Preleva minuti.
1335	JSR $1353	Traduce in Ascii
1338	STA $0403	e stampa su
133B	STX $0404	video.
133E	STY $0405	Separatore.
1341	LDA $DC09	Preleva secondi.
1344	JSR $1353	Traduce in Ascii
1347	STA $0406	e stampa su
134A	STX $0407	schermo.
134D	LDA $DC08	Restart.
1350	JMP $FA65	Torna a IRQ.
Traduzione	**da BCD**	**in codici**
Schermo	**(valida anche**	**per ASCII)**
1353	PHA	Salva il byte.
1354	AND #$0F	Esclude nibble hi.
1356	ORA #$B0	Somma base codici.
1358	TAX	Conserva valore.
1359	PLA	Recupera byte.
135A	LSR	Quattro rotazioni
135B	LSR	a destra per spo-
135C	LSR	stare nibble alto
135D	LSR	in nibble basso.
135E	ORA #$B0	Somma base codici.
1360	RTS	Return.

Commodore
COMPUTER
CLUB
La rivista degli utenti di sistemi Commodore

recensioni

AMIGA
Commodore MODEL 1081
POWER

AFTER THE WAR

Nonostante Bush e Gorbaciov si diano tanto da fare per ridurre gli armamenti, c'è chi scrive ancora giochi ambientati in città lasciate distrutte e selvagge da una guerra nucleare totale, ricalcando magari il mitico Kung Fu Master.

Il gioco

Jonathan Rogers è un sopravvissuto all'Olocauso nucleare. Con i suoi jeans, scarponi e bicipiti deve raggiungere una piattaforma di lancio sulla quale lo attende un missile per portarlo alle colonie spaziali scampate al disastro.

Ovviamente la base di lancio è all'interno di una fortezza ben protetta da robot con sembianze di viventi più o meno orrorifici: pantegane (="toponi", per i non lombardi) volanti, canguri pugilatori, punk vigliacchi, tori inferociti senza vedere il rosso, cani affamatissimi, eccetera.

Jonathan può comunque reagire con ben venti tipi di mosse di attacco e di difesa: da calci a tre livelli a pugni a due livelli, dal balzo acrobatico con calcagnata sulle mascelle del nemico, allo sgambetto (scherzo...). Inoltre può afferrare armi e bombe sparpagliate per le scene di gioco in numero (molto) limitato.

Il programma prevede due caricamenti separati: chi ha già superato i primi tre livelli ottiene una parola d'ordine con la quale può successivamente partire direttamente dal primo degli altri due livelli.

Picchiaduro guerrafondaio ma dotato di buona grafica

Computer: Amiga, C/64
Gestione: Joystick
Tipo: Arcade picchiaduro
Softhouse: Dinamic

La tecnica

Le scene di sfondo, pur se certamente non originali nè più realistiche grazie al disarmo incombente, sono ben fatte e molto realistiche.

Si passa dal ponte di Brooklin ai garage, sempre con ottime scelte di colori e dovizia di dettagli. Le sequenze di movimento del protagonista sono ottime, un po' meno quelle dei nemici, che però sono numerosissimi e molto scenografici. I suoni sono pochi, ma buoni.

Il voto

Non troppo originale, ma tecnicamente decisamente valido, pur se con pochi incentivi. 8.

ANT HEADS

Ritornano i formiconi giganti di cui ci siamo già occupati sul n. 74

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Adventure/Arcade
Softhouse: Cinemaware

Il seguito di **It Came From the Desert** è un vero e proprio remake con pochissmie modifiche nelle scene e la stessa eccezionale qualità Cinemaware.

Il gioco

Sono passati cinque anni da quando i formiconi giganti hanno terrorizzato per la prima volta la pacifica città di **Lizard Breath**. Nei panni del dottor **Greg Bradley** si dimostrò la loro esistenza e si convinse tutti a combatterle vittoriosamente (se avete completato il gioco).

Sfortunatamente un gruppo di formiche giganti trovò una scorta di plutonio radioattivo e, racchiusolo in recipienti di metallo, lo portarono alla regina madre. Ora hanno un nuovo nido nascosto, una nuova regina più potente di quelle di prima, un depostio segreto di uova e tante cattive intenzioni.

Oltretutto hanno trovato il modo di ridurre in schiavitù alcuni abitanti della città, che di quando in quando si trasformano, alla Visitors, in esseri ibridi dalla testa di formica, da affrontare a colpi di Colt 45.

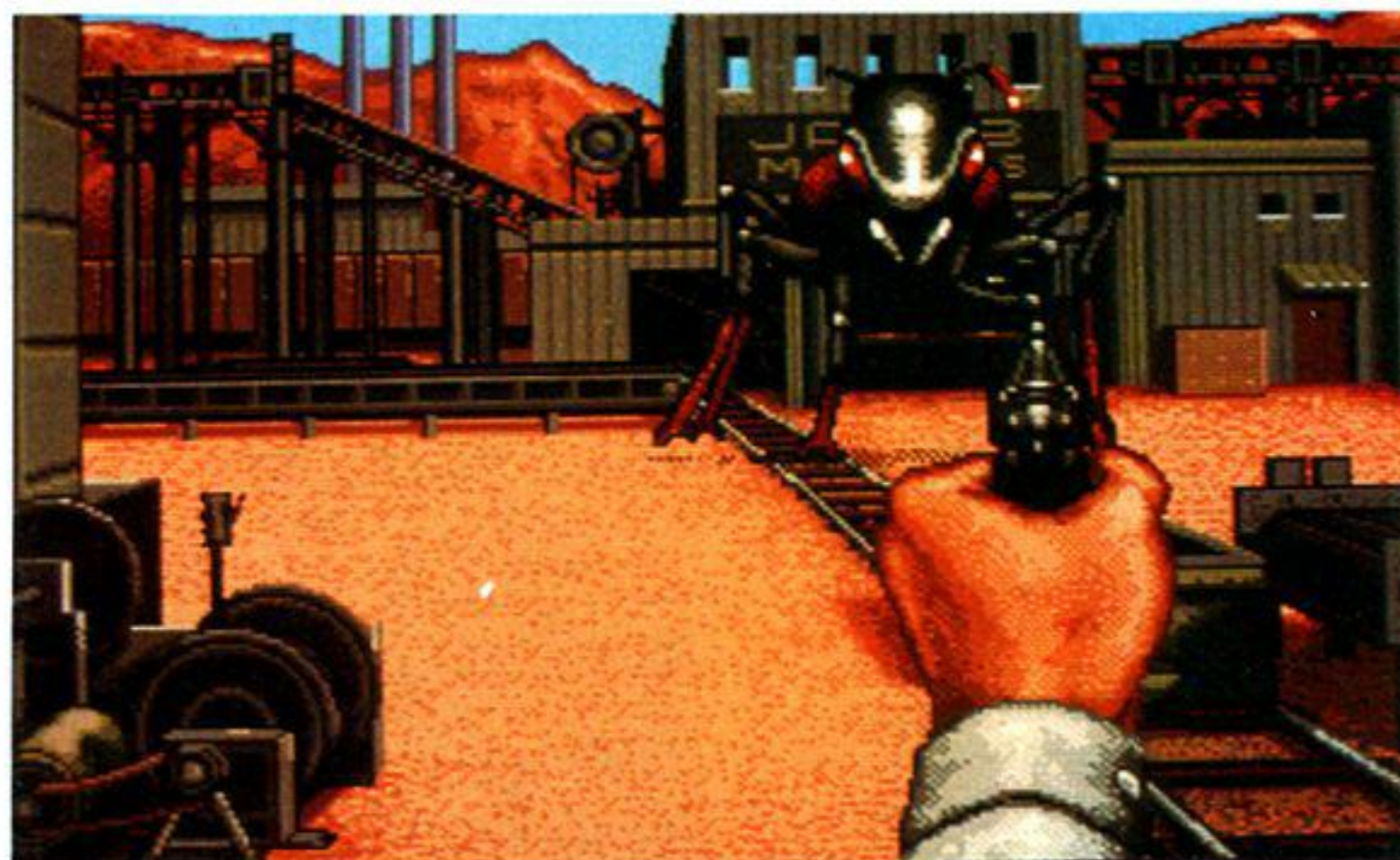
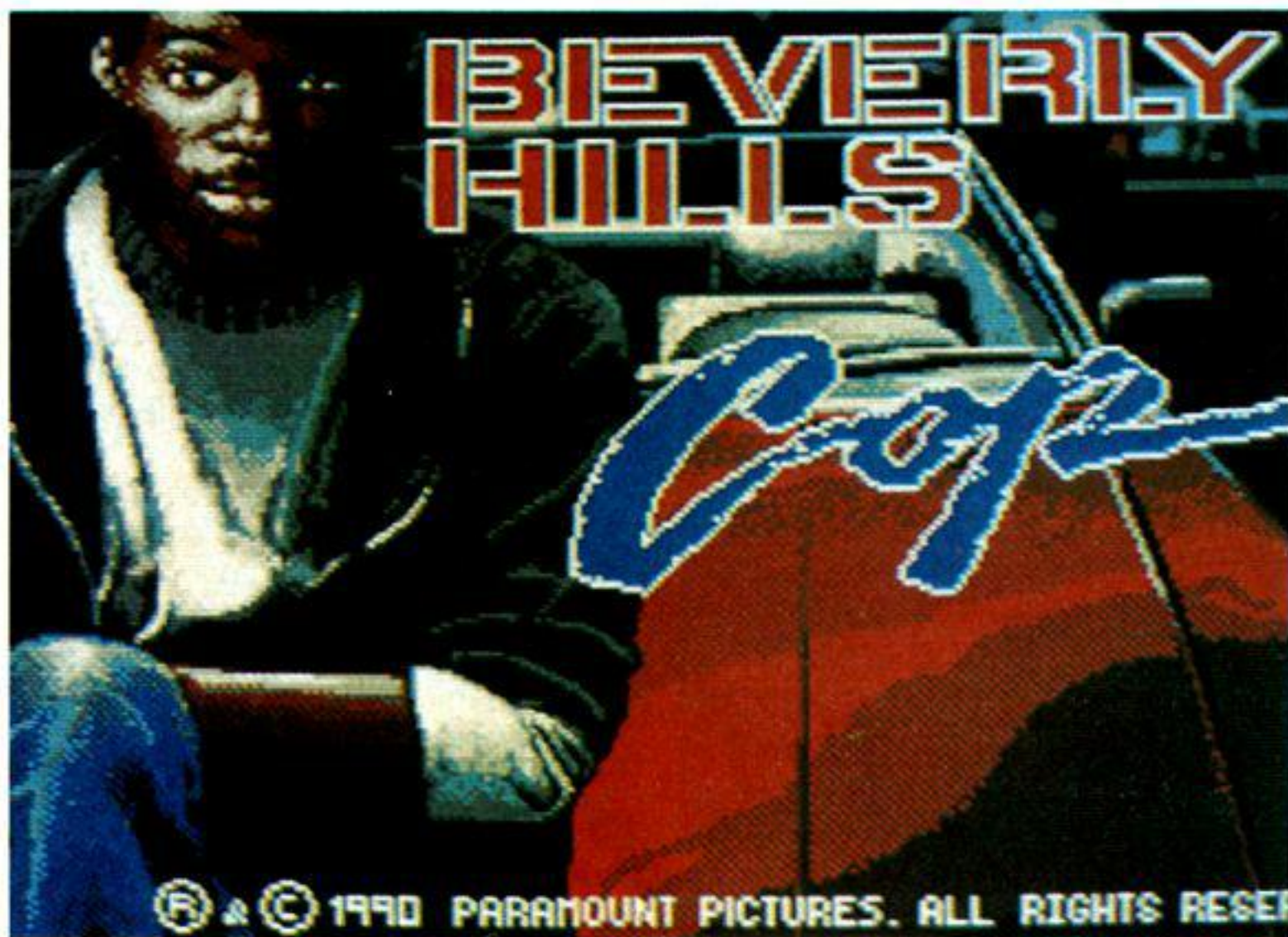
La scena di gioco è la solita Lizard Breath, con Ice, Dusty, Biff, il dottor Wells e molti dei personaggi già incontrati cinque anni prima. La differenza è che ora sono sconosciuti e diventa molto difficile sapere quale sarà la prossima mossa da fare.

La tecnica

Essendo questo programma una specie di "upgrade" in due dischi di It Came From the Desert, la gran parte degli effetti sonori e della grafica è risulta conservata identica alla precedente versione.

Le modifiche principali sono alcune opzioni supplementari di gioco ed una maggiore verbosità e profondità dei dialoghi. Ad esempio, la stampa fornisce ad ogni mossa delle notizie che possono sembrare inutili, ma raramente lo sono effettivamente. La qualità, insomma, è ai massimi livelli sia per quanto riguarda le (comunque semplici) animazioni, sia per le scene, sia per la musica e gli effetti sonori, sia per la trama, sia per l'atmosfera anni '50.

Il voto

Non ci sono molti dubbi, una volta accettato che è un remake di un gioco già fatto. 9.

BEVERLY HILLS COP

Chi ha visto il divertente film con l'attore **Eddie Murphy** non si illuda, difficilmente potrà sollazzarsi altrettanto con questo programma.

Il gioco

Come nel film, **Axel Fooley**, il detective americano simpatico e sboccatissimo, è sulle tracce di una banda di contrabbandieri di armi. In questo caso il film non è in due "tempi", ma in quattro, tante sono le differenti scene previste dal programma. Nella prima si deve raggiungere la banda in un capannone correndo in prospettiva a 30 gradi tipo PaperBoy (su quattro ruote, però). Curioso è che per uccidere un cattivone si deve premere il pulsante due volte: una prima volta si prende la mira, la seconda si esplode il colpo vero ed proprio.

Nel secondo "tempo" si inseguono dei furgoni carichi delle armi contrabbandate e bisogna cercare di colpire i nemici evitando di impallottolare le macchine di innocenti conducenti frapposti. Nelle successive due sezioni si ripete vagamente la trama della prima, come in **The Untouchables**, dovendo uccidere i contrabbandieri e, finalmente, Mister Big, con una prassi simile a quanto visto nella prima fase, ma con scene differenti e una vita molto più dura tra i tanti nemici.

La tecnica

La grafica è piuttosto scarsa, fatta salva la schermata iniziale con Eddie Murphy. Gli effetti sonori sono patetici e l'animazione, a parte la seconda fase, decisamente insufficiente.

Il voto.

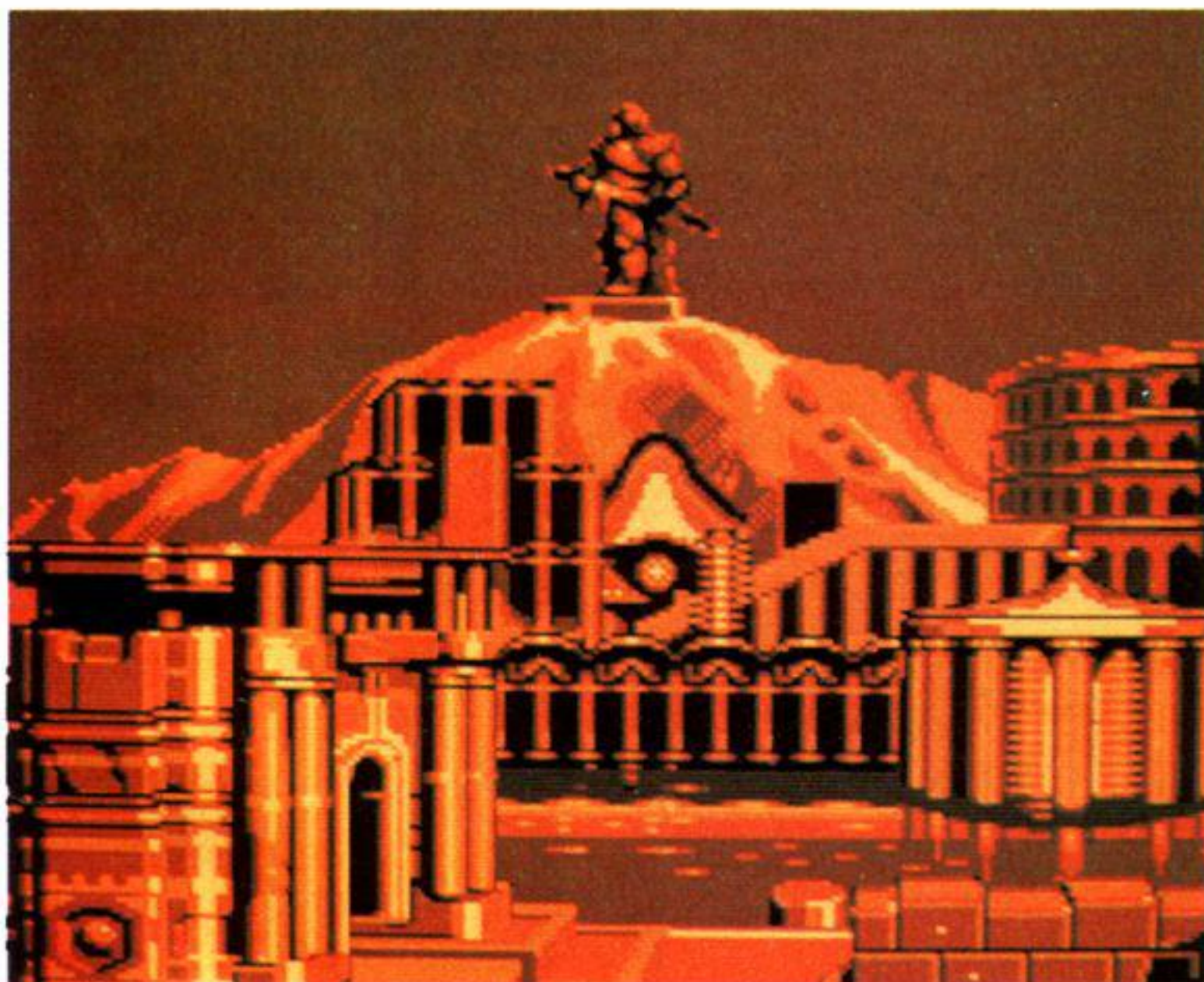
Insufficiente, specie considerando che il gioco costa ben quarantamila lire. 5.

Dal grande schermo al video del nostro Amiga. Ma ci si diverte di più al cinema

Computer: Amiga, C/64
Gestione: Joystick
Tipo: Arcade scorrevole
Softhouse: Tynesoft

BLACK TIGER

Combattere ed uccidere dragoni è lo scopo principale del protagonista

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Arcade a piattaforme
Softhouse: US Gold

L'ultima conversione della Capcom da parte della US Gold, dopo **Strider** e **Ghouls and Ghosts** vede in azione un guerriero che spara raffiche di mazze ferrate come se fossero raggi laser.

Il gioco

La **Tigre Nera** del titolo è il personaggio controllato col joystick: un guerriero leggendario, conosciuto per la sua terribile forza ed il suo eroismo. La missione è medievale: scovare e distruggere tre dragoni che, provenienti dalle fiamme dell'inferno, minacciano il mondo.

Mister Tigre è in grado di camminare lungo le piattaforme decoratissime ed impervie che costituiscono la scena, scalare colonne, compiere balzi e salire scalinate. Ogni scena di gioco vede impegnate torme di creature diaboliche che non hanno niente di meglio da fare che cercare di ucciderci senza provocazione. Il protagonista vede diminuire la propria carica vitale ad ogni colpo subito, ma può difendersi lanciando mazzi di asce e spade e facendo buon uso di corazza e scudo. In giro per le piattaforme si possono trovare: armi speciali (adatte per varie qualità di mostri), energia extra, incantesimi ed oggetti vari. Le chiavi, ad esempio, possono aprire forzieri. Alla fine di ogni livello si trova, indovinate, il drago di fine livello da uccidere ovviamente (non capita mai che si debba portargli dei fiori o delle opere buone...), od anche dei guardiani altrettanto antipatici.

La tecnica

Non si tratta della grafica più eccitante che abbiamo visto su Amiga.

L'area di gioco è piuttosto ristretta, mentre lo sfondo, dotato di colori molto stridenti, scorre in modo assai poco fluido.

Sono invece gradevoli alcune semplici sequenze animate, come l'esplosione dei draghi, sebbene contribuiscano a confondere la vista durante il gioco. I suoni sono sufficienti, ma nulla di più.

Il voto

Un gioco gradevole per gli amanti delle piattaforme, costa solo diciottomila lire, ma non è nulla di eccezionale. 6 1/2.

CONQUEROR

Un programma originalmente per **Archimedes**, vagamente simile a **Virus** e **Zarch** per tipo di grafica, trasportato con ottimi esiti sul nostro beniamino a 16/32 bit.

Il gioco

Prima di giocare si deve scegliere se controllare un panzer americano, tedesco o russo, tenenendo presente che non è possibile mettere contro americani e russi, perchè nella seconda guerra mondiale erano alleati.

Si deve anche scegliere il tipo di controllo da usare, ed è consigliabile scegliere la tastiera se si ha a disposizione un solo joystick. Se si desidera giocare con (non "contro"!) un compagno, si può sfruttare la possibilità di fare manovrare la torretta di fuoco, ricalcando così il realistico doppio ruolo di manovratore e cannoniere.

A disposizione si hanno tre diverse modalità di gioco: **Arcade, Attrition** e **Strategy**. Nel primo si hanno tre vite, si inizia col carrarmato più leggero e si devono semplicemente cannoneggiare i tank nemici, acquisendo punti per ogni rivale distrutto. Procedendo, i nemici diventano sempre più abili e numerosi, ed il tank sempre più potente e difficile da manovrare.

Il modo **Attrito** consente invece di giocare un misto di arcade e strategia, dove si inizia con cinque tank di differente potenza ed il computer ne sceglie uno inferiore. Ovviamente si controlla un solo carro per volta, mentre agli altri si possono dare ordini grazie ad un sistema di mappe e segnali radio.

Nella modalità strategica si deve distruggere completamente il nemico, o mantenere il controllo di una zona selezionata per almeno un minuto di tempo, compito assai arduo data la vastità delle zone.

Inoltre si possono acquistare o vendere carrarmati, inzialmente sedici, per raggiungere l'assetto tattico dettato dal nostro fiuto strategico.

Da Archimedes ad Amiga; senza rimpiangere nulla del primo computer

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Arcade
Softhouse: Rainbow Arts

La tecnica

La routine di **landscaping**, ovvero quella che genera la prospettiva di gioco, è di **David "Elite" Braben**, che come si nota dal soprannome, è il padre del leggendario programma Elite, autore di Virus, la conversione di Zarch effettuata per Archimedes. I movimenti sono decisamente fluidi, e anche se la nitidezza grafica non è esaltante, è certamente caratteristica e suggestiva. I suoni sono pochi, limitandosi a qualche esplosione e ronzio di torretta, ma sono realistici.

Il voto

Un best seller all'estero, certamente lo sarà anche in Italia. 9.

DYTER 07

Si tratta di salvare alcuni professori in vacanza su diverse isole(!)

Computer: Amiga inespanso,
Gestione: Joystick
Tipo: Arcade
Softhouse: ReLine

L'autrice di **Hollywood Poker Pro**, il migliore strip poker per computer, non è molto famosa, forse perchè produce normalmente programmi non eccellenti, ma sicuramente interessanti. Questo è sulle orme dei vari **Silkworm, Choplifter** e **Blue Thunder**.

Il gioco

La scena è rappresentata inizialmente da isole tropicali, dove inizia la nostra missione che ci porterà a distruggere dei robot nemici. Il nostro mezzo di locomozione, e di lotta, è un elicotterino molto armato, equipaggiato anche di un robot sottomarino, che fornisce chiaramente il pretesto per giocare una specie di gioco nel gioco.

Durante la missione si devono raccogliere, stile Defender, alcuni professorini che sono stati isolati (così imparano) su isolette deserte.

Si noti che, a dispregio della scienza, il gioco può essere terminato senza sal-

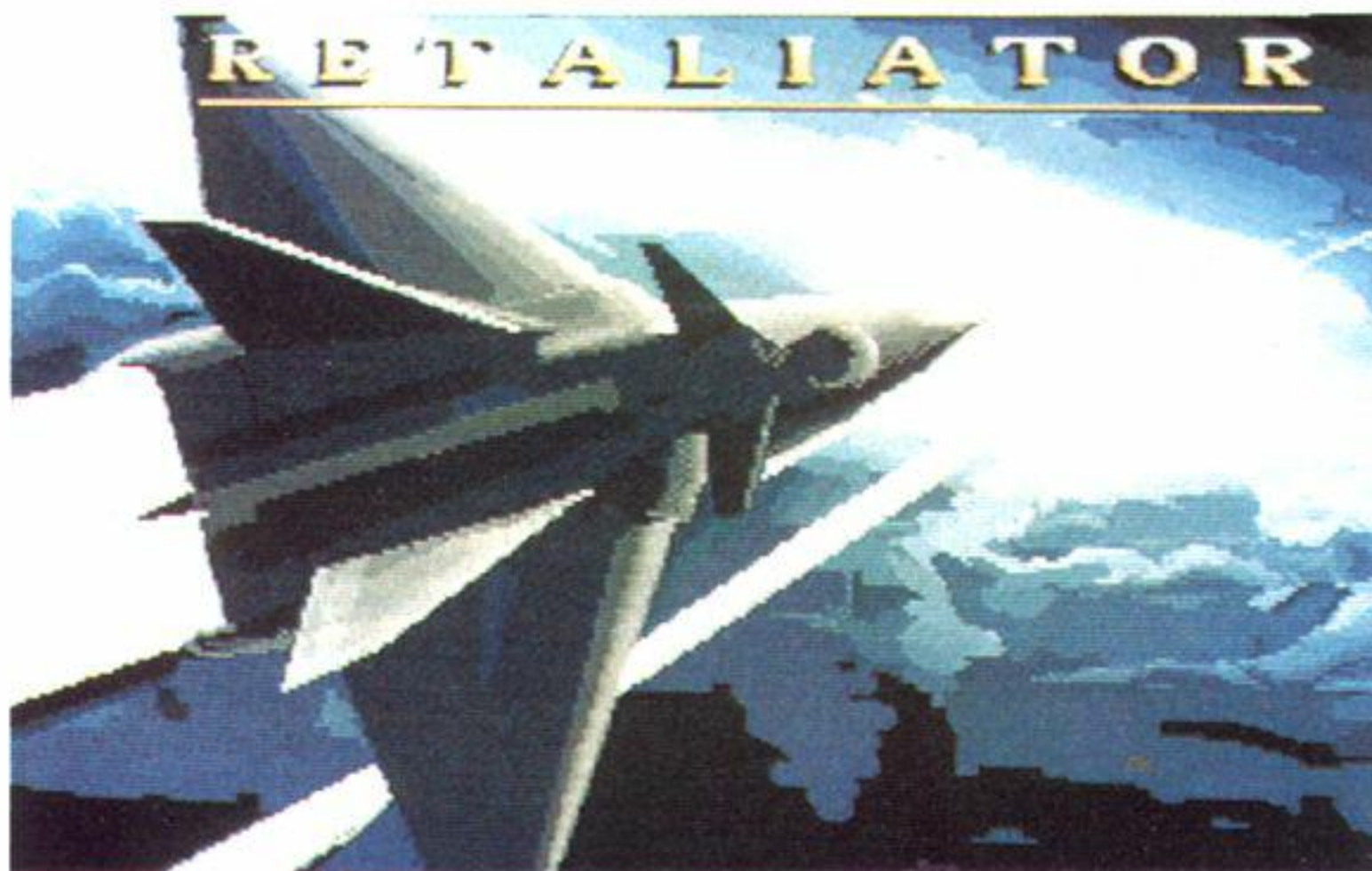
vare i professori, anche se si ottiene un punteggio più basso.

La tecnica

Le scene sono molto colorate, con particolari grafici piccoli ma dettagliati sia nelle scene di immersione sia in quelle di emersione. Gli effetti sonori sono decenti.

Il voto

Per gli amanti degli arcade "spara a tutto" un pezzo di innegabile interesse; per gli altri videogamers un titolo tra tanti. 7+.

F-29 RETALIATOR

Il gioco

Si inizia scegliendo il "grado" del giocatore: da luogotenente a colonnello. Poi si deve scegliere uno scenario di operazione oppure partire direttamente con la missione di guerra (in codice: Zulu Alert). Infine si deve scegliere l'aereo: un **F-29** o un **F-22**, che permarrà per tutta la carriera (salvo scegliere inizialmente un nome diverso per l'archivio salvato su dischetto dopo ogni sessione e ripetere le scelte). Ovviamente è consigliabile iniziare con una prova di volo, per familiarizzare. Sono previste dieci missioni di prova in Arizona, per colpire pseudo bersagli disseminati in un'area di oltre mille miglia quadrate: veicoli, ponti, stabilimenti, basi, eccetera. Il nemico per eccellenza della fase di test è comunque il **Mig- 29** teleguidato, in grado di simulare una battaglia mozzafiato. Per quanto riguarda i controlli, si può avere visione dell'esterno (da varie angolazioni) e da un ipotetico satellite. Sono previsti il pilota automatico per inseguire il nemico a distanza, diversi tipi di radar per bersagli volanti e terrestri, mappe animate oltre ad una quantità di armi: quattro tipi di missili aria-terra e cinque tipi di missili aria-aria, caricabili ad inizio missione tramite apposite sequenze di armamento.

Un grande simulatore di volo da una delle più rinomate softhouse

Computer: Amiga inespanso
Gestione: Joystick, tastiera
Tipo: Simulatore di volo
Softhouse: Ocean

Le missioni di guerra vera e propria sono varie e disparate e consentono di dare lustro alla propria carriera salvata su dischetto: abbattere mig, fare saltare stabilimenti bellici o pozzi petroliferi, bombardare un'arma segreta super corazzata partendo da una portaerei ed altro ancora.

La tecnica

Ineccepibile il realismo delle vedute, in particolare delle navi, delle portaerei e delle isolette, nonchè delle animazioni, in ottima prospettiva, anche se le sagome sono necessariamente piuttosto squadrate per essere mosse con sufficiente velocità. La vista del "cruscotto" dell'aereo non è molto realistica, essendo poco "affollato", ma si devono premere appositi tasti per accedere ai numerosi strumenti e schermi di lavoro predisposti. Gli effetti sonori sono ottimi, sebbene sia necessario collegare un'amplificatore esterno per avere un'idea della qualità.

Il voto

Una delle migliori simulazioni in circolazione, non solo per Amiga. 9

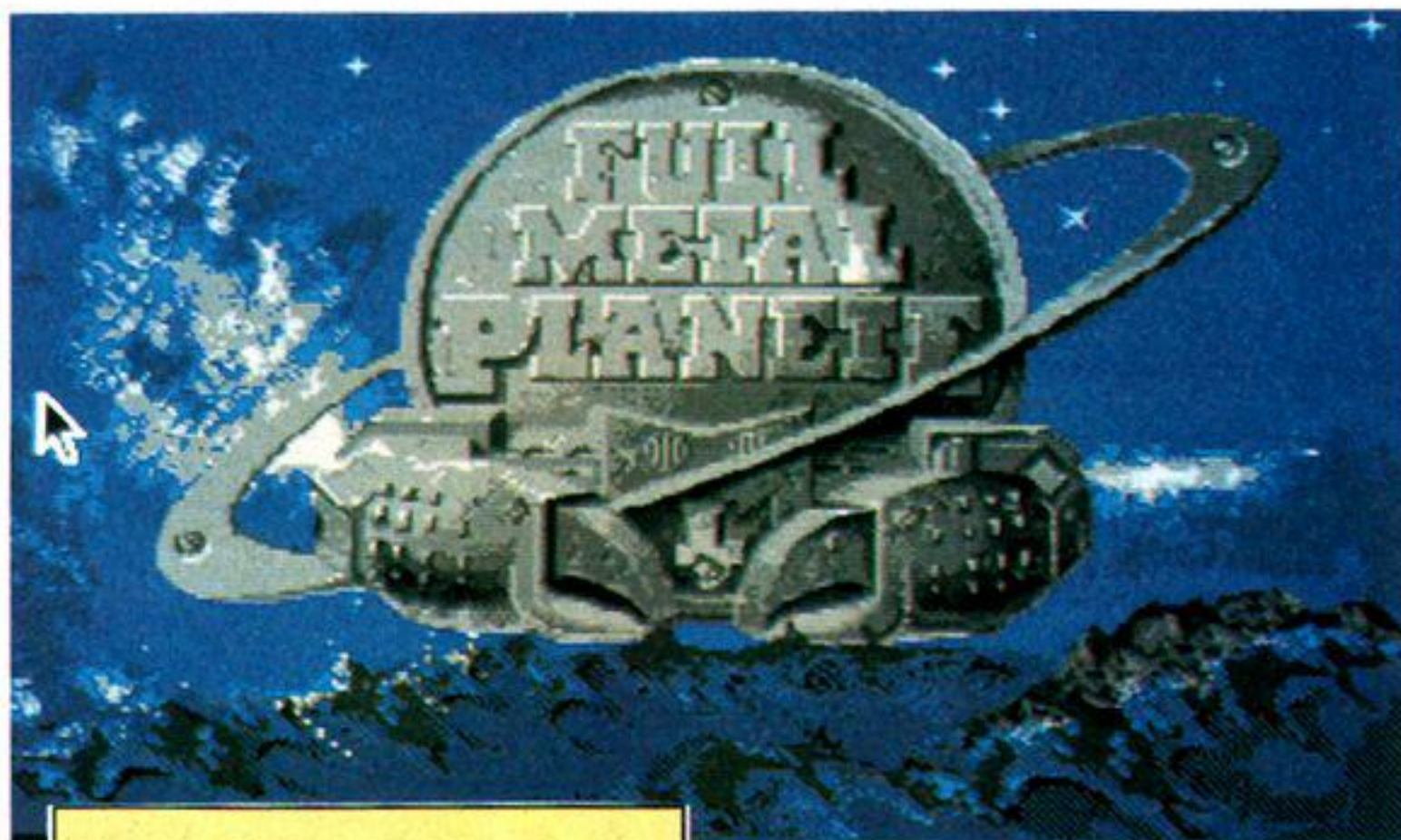
FULL METAL PLANETE

Da una a quattro basi volanti sono approdate su un pianeta che, come dice il nome, è tutto d'acciaio (forse è un espediente per semplificare la grafica di sfondo...). Ogni giocatore, eventualmente simulato da Amiga, deve raccogliere una quantità di prezioso metallo disseminato tra gli anfratti; per farlo si deve combattere contro i nemici in 25 mosse, cercando di racimolarne quanto più possibile. Si gioca a turno, un po' come nei gochi tipo **Risiko**. Inizialmente si decide dove sistemare la nave madre, poi i propri armamenti, rappresentati da cinque carrarmati, due navi e due strani pezzi di metallo chiamati Crab e Hen. Il primo è quello che raccoglie il materiale prezioso per portarlo sulla nave, il secondo può fare il lavoro di Crab, ma anche costruire nuovi pezzi di macchine, utilizzando il materiale raccolto.

I movimenti degli armamenti sono limitati sia da un numero massimo di mosse per ogni turno del giocatore (inizialmente sono cinque punti di manovra, per diventare 25 nelle fasi finali), sia dal tipo di terreno. Ad ogni turno, infatti, parti del pianeta vengono sommerse, altre emergono, e bisogna stare attenti a non fare inghiottire o lasciare isolati i propri pezzi.

Quando si hanno sufficienti punti di manovra, dopo le primissime mosse, si può attaccare i mezzi avversari. Per farlo si devono avere due propri pezzi nell'area di fuoco, tenendo conto che ogni pezzo non può sparare più di due volte per mossa. Il vincitore è il giocatore, umano o mosso da Amiga, che riesce a sopravvivere e ad avere il maggior numero di materiale prezioso e di pezzi.

La tecnica

I sei livelli di aggressività dei ruoli giocati dal computer contribuiscono ad aumentare la varietà e la piacevolezza del gioco. In effetti sono previsti anche stili di gioco molto remissivi, che consentono di impratichirsi senza prendere troppe batoste alle prime partire, per poi passare a spietati aguzzini che fanno a pezzi le nostre truppe prima che si faccia in tempo a dire "mamma mia".

Programma strategico con grafica da arcade in grado di impegnare sino a quattro giocatori

Computer: Amiga, C/64
Gestione: Mouse, Joystick
Tipo: Arcade strategico
Softhouse: Infogrames

Il pianeta è visto dall'alto e la grafica è piacevole, senza essere nulla di strepitoso, ma comunque appropriata al tipo di gioco, chiaramente più strategico che arcade. Gli effetti sonori sono scarsetti.

Il voto

Un gioco tattico abbastanza originale, supportato da una realizzazione tecnica efficace e senza troppe ricercatezze. 7.

KID GLOVES

Addormentarsi è facile; il difficile è svegliarsi dall'incubo in cui siamo caduti

Computer: Amiga inespanso
Gestione: Joystick, Tastiera
Tipo: Arcade a piattaforme
Softhouse: Millenium

Un gioco a piattaforme, dove ogni schermo racchiude dei puzzle, sempre uguali, da superare con intuito e tempismo, stile **Miner 2049er**.

Il gioco

Kid deve ritornare a casa nell'incubo che sta sognando dopo essere penetrato nello studio di un suo zio avventuriero.

Per farlo, deve superare cinque zone temporali, iniziando da una giungla preistorica per proseguire nell'antico Egitto, nell'era glaciale, nella Londra della rivoluzione industriale e nella psichedelica, modernissimaa, West Coast.

Ciascuna zona è suddivisa in dieci schermi, contenenti una quantità di mostri da evitare, oggetti da raccogliere, porte da raggiungere e piattaforme sulle quali balzare agilmente.

I mostri in circolazione possono facilmente essere abbattuti da un colpo dell'arma in possesso di Kid: inizialmente una fionda con due monete riutilizzabili (se i mostri si fossero chiamati Alemaos, il gioco sarebbe sicuramente di provenienza italiana ed ambientato a Bergamo...), bombe, laser, stelle d'acciaio rimbalzanti, eccetera.

Non mancano alcuni piccoli incantesimi, in numero molto limitato e difficilmente reperibili nel gioco, che servono per aprire porte, neutralizzare ostacoli pericolosi e surgelare i movimenti di mostri. Se si riesce a terminare l'avventura, il piccolo ritornerà felicemente a casa.

La tecnica

La grafica di Kid Gloves è stata chiaramente studiata per essere colorata e nitida, con un sapore leggermente infantile, come suggerisce la trama del gioco. Gli sfondi sono molto belli, in particolare quelli egiziano e californiano, molto difficili da superare a causa del rapido movimento dei nemici. Gli effetti sonori sono molto buoni, mentre la musica ricorda troppo quella di un C/64 per essere apprezzabile in Amiga.

Il voto

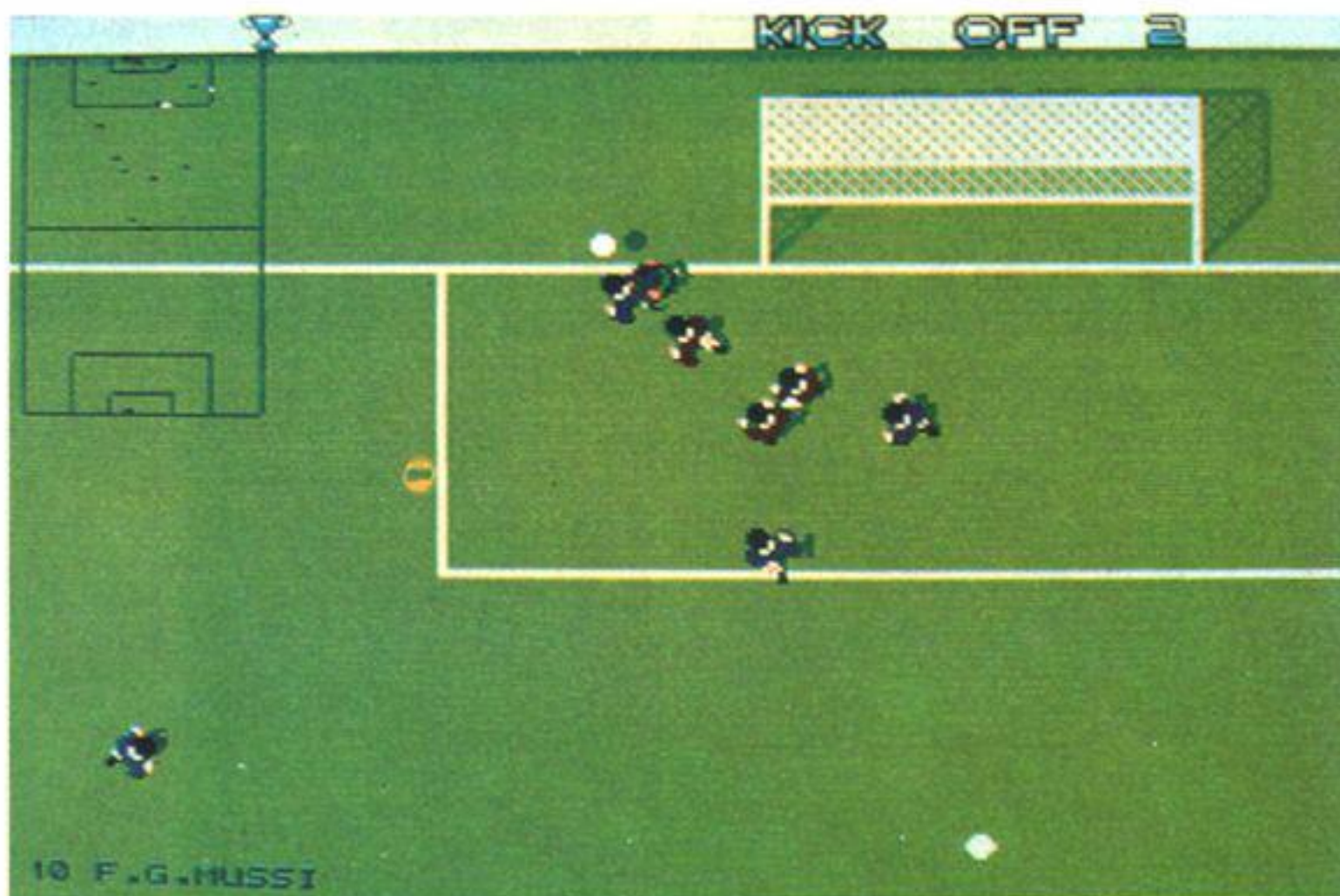
Un gioco senza troppe pretese, ma onesto. 6+.

PLAYER MANAGER

Kick Off di Dino Dini è, a ragione, considerato uno dei migliori "Soccer Games" in circolazione per Amiga. Ora, dallo stesso autore, è stata prodotta questa nuova versione, che aggiunge la possibilità di gestire la formazione. In pratica si tratta del "solito" Kick Off, con le fasi di movimento viste dall'alto velocissimamente e con un paio di aggiunte molto importanti.

Il gioco

Ciascun giocatore è condizionato da due parametri: livello energetico, o **stamina** ed infortuni. Ciò significa, ad esempio, che un attaccante con poca energia può avere parecchie difficoltà contro un difensore in buona forma, e che può subire più facilmente infortuni che lo rendano inutilizzabile per più partite. Ciascuna squadra prevista gioca in modi diversi ed è composta da giocatori di differenti abilità, come riportato da apposite schermate statistiche.

In pratica, si può giocare in due ruoli. Il primo, più semplice, è un vero e proprio ruolo da videogiocatore, dove si controlla con il joystick l'intera squadra, un giocatore alla volta. Il secondo ruolo prevede la possibilità di rivestire i panni di un solo giocatore, controllando però, come allenatore, la formazione dei giocatori che scendono in campo. In questo caso il computer manovra tutti i giocatori, tranne uno, con abilità proprozionale al valore ed allo stato di forma di ciascun elemento.

AMettetevi al comando di una squadra di calcio

Computer: Amiga inespanso
Gestione: Joystick
Tipo : Arcade sportivo
Softhouse: Anco

Non si pensi che questo spostamento da puro arcade sportivo a arcade/gestionale abbia comportato sacrifici in uno o nell'altro aspetto: la grafica al fulmicotone e la fluidità di manovra delle scene di gioco è identica, mentre i numerosi schermi di dati statistici, dettagli di trasferimenti e di campionato, danno un aspetto serissimo al programma anche dal punto di vista "strategico".

La tecnica

La grafica è sempre la stessa, quindi estremamente veloce, fluidissima e colorata. Gli effetti sonori sono gradevoli, anche se abbastanza scarsi di numero. La lingua è italiana, senza troppi errori grammaticali(...).

Volendo fare i pignoli, si sarebbe potuto curare un poco di più il "look" delle schermate delle fasi manageriali, piuttosto piatte ed asettiche.

Il voto

Un programma eccellente, da collezionare senza dubbi. 9.

PURSUIT TO EARTH

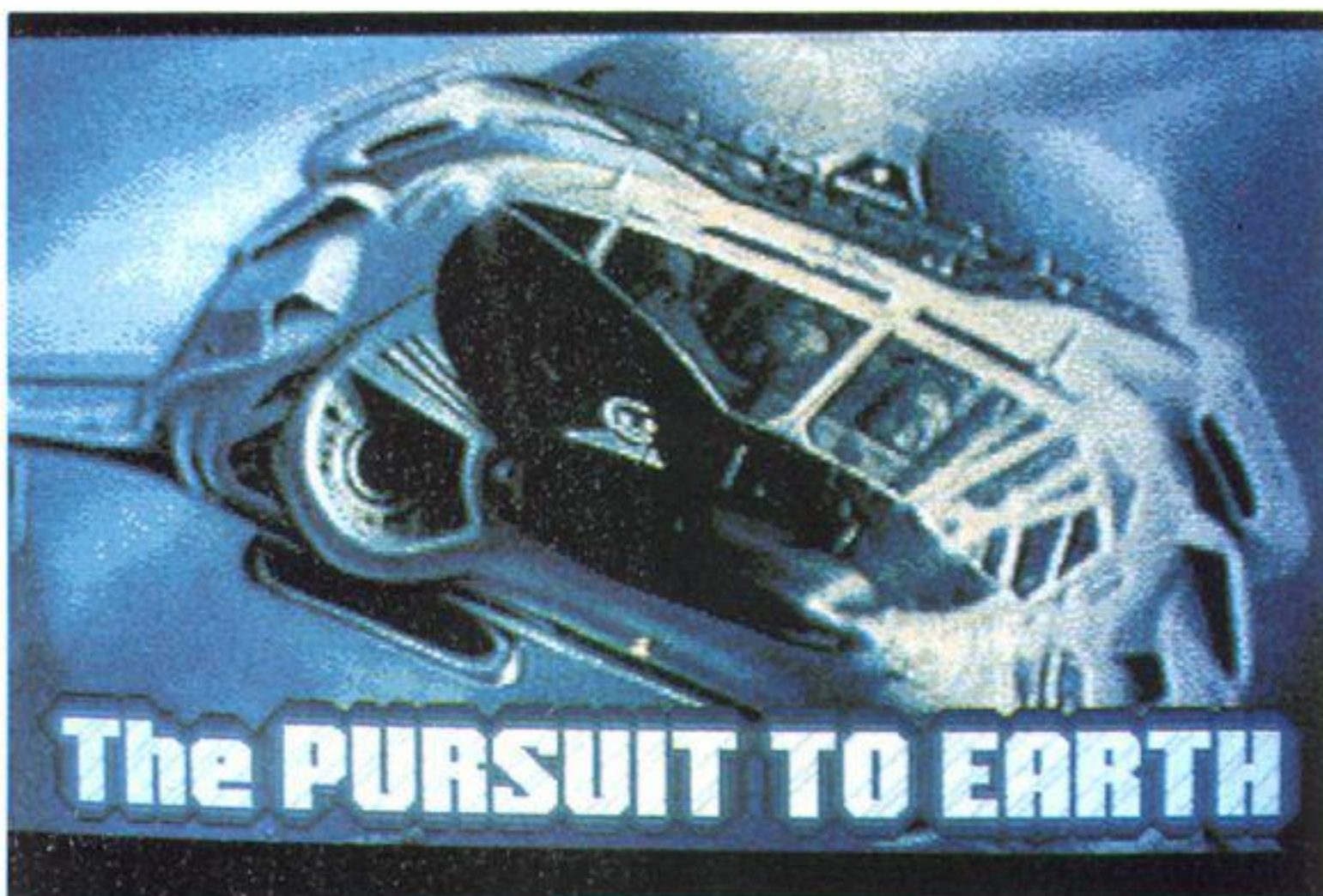
Un videogame che vale meno del dischetto su cui è registrato

Computer: Amiga inespanso
Gestione: Joystick
Tipo: Arcade spaziale
Softhouse: Exocet

Un gioco sullo stile del vecchissimo **Gyruss**, dove si viaggiava sul bordo dello schermo, di pianeta in pianeta, per disintegrare gli alieni.

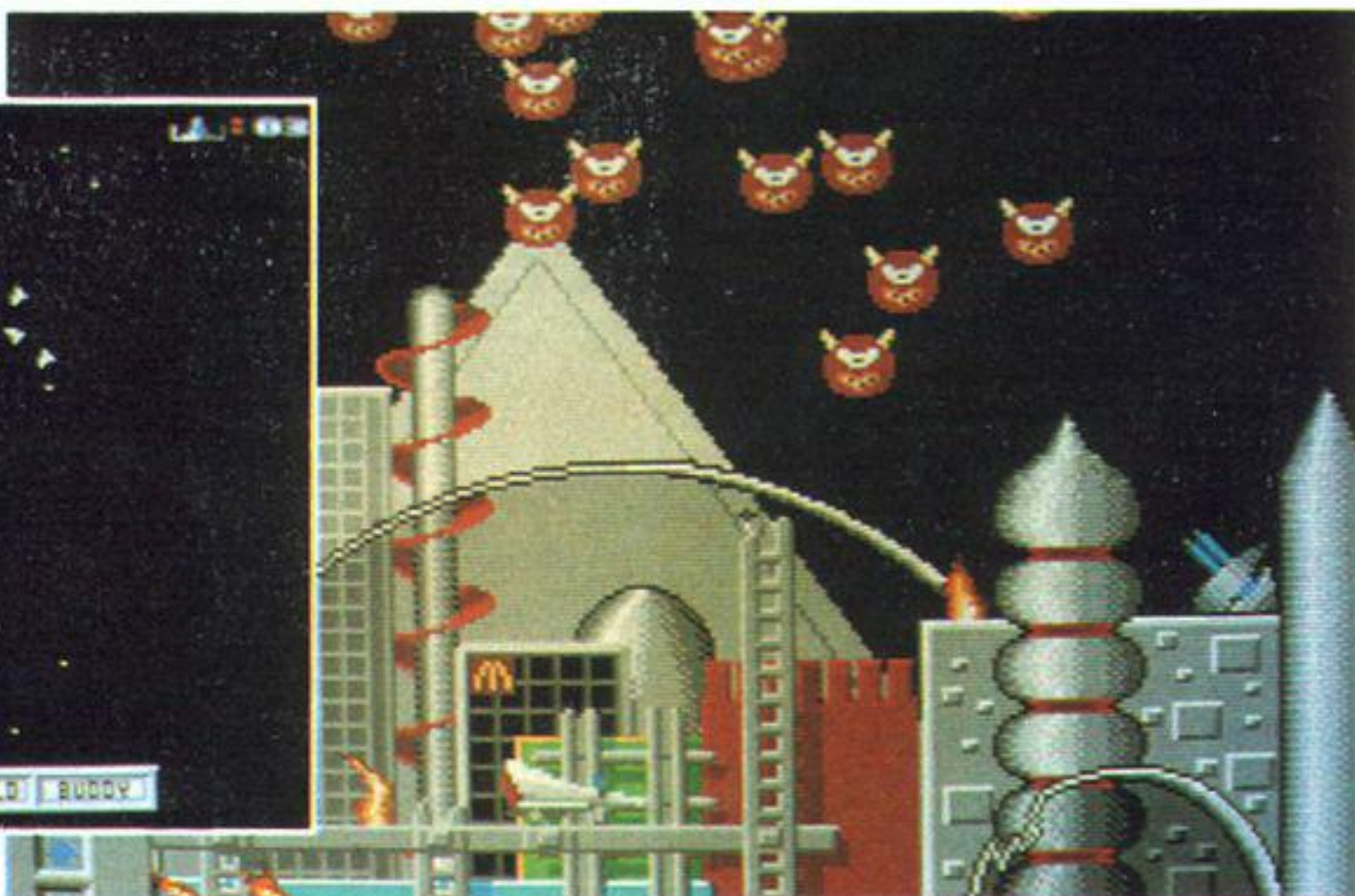
Il gioco

La trama è la stessa di Gyruss: le scene sono i pianeti del sistema solare, attaccati dalle forze aliene.

La destinazione è sempre la stessa, la madre terra, tranne che del pianeta **Platblat** non avevamo mai sentito parlare. La trama è comunque estremamente povera e scarsa di incentivi.

La tecnica

La grafica è molto simile a quella del vecchio videogioco **Arcadia**, con gli incentivi e le sofisticazioni permesse dai sedici bit. E' piuttosto colorata, molto veloce, ma quasi assolutamente priva di animazione. Inoltre la velocità diminuisce con l'aumentare degli alieni sullo schermo. Il suono è di scarsissima qualità, soltanto qualche effetto modesto sonoro, male digitalizzato, ed una colonna sonora stressante.

Il voto

Decisamente insufficiente, un game da dimenticare in fretta. 4.

A bordo della nostra navetta spaziale andiamo a caccia di alieni

Computer: Amiga inespanso
Gestione: Mouse
Tipo: Arcade spaziale
Softhouse: Activision

WARHEAD

Un tipico arcade spaziale con grafica 3D pseudo vettoriale come il glorioso **4D Time War** del vecchio C/64 (zilioni di volte più bello).

Il gioco

Tempo: futuro. La Terra ha subito un duro attacco da alieni a forma di insetto provenienti da Sirio.

I motivi sono sconosciuti, ma è chiara la prevalenza tecnologica del nemico, sebbene tutte le speranze siano riposte nel battello spaziale d'attacco chiamato FOE-57.

La giornata tipica di un pilota del FOE-57 inizia con la lettura del giornale delle missioni: nel nostro caso, le prime operazioni consentiranno di familiarizzare con le manovre di base, compreso l'attraccaggio con la base spaziale, molto delicata.

Dopo questo tirocinio, si può incominciare ad usare il salto iperspaziale, per portarsi nelle zone del sistema solare dove occorre la nostra opera, che ovviamente bisogna pattugliare con motori normali.

Tutto il gioco è visto attraverso la finestra frontale della navetta, che dà una visione molto confusa alle alte velocità. Per fortuna si dispone di un certo numero di piloti automatici che possono aiutarci, per esempio, a prendere contatto automaticamente con le navi nemiche.

Le missioni vanno da semplici ricognizioni all'attacco di orde nemiche.

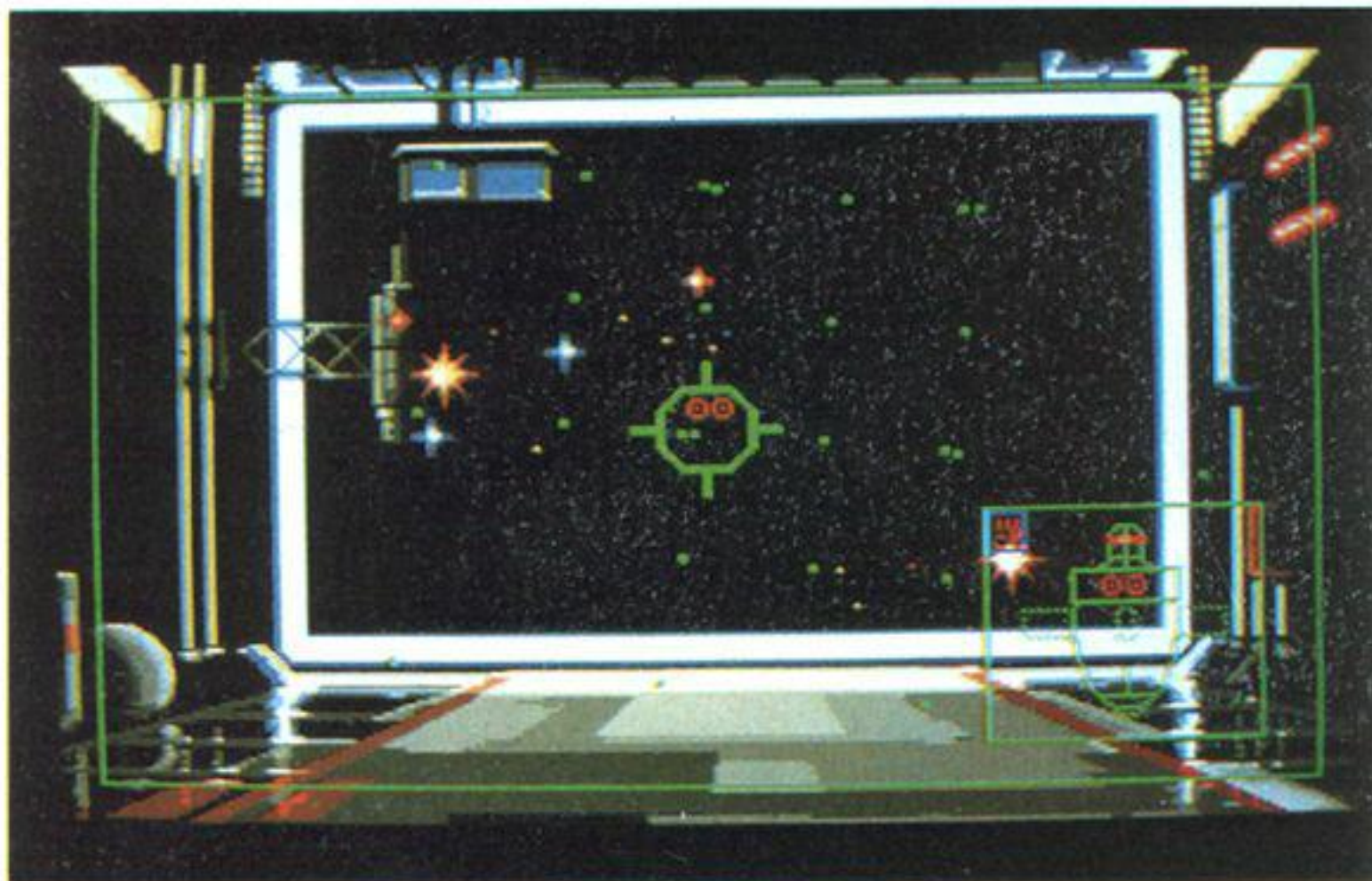
Vi sono anche delle sorprese, come ad esempio il Berzerker, che odia tutte le forme viventi, compresi i nemici e noi.

La tecnica

La musica di introduzione è molto suggestiva e di effetto. Gli effetti sonori sono ridotti, ma piuttosto appropriati.

La grafica in prospettiva tridimensionale è gradevole, fluida e molto colorata. Per quanto riguarda la parte manageriale, sono curati i particolari delle informazioni sui pianeti: sistemi solari, orbite planetarie, eccetera.

Il voto

Un gioco curato, molto coinvolgente, tecnicamente valido. 8.

AMIGA 3000

Facendo uno strappo alla tradizione (che ci vede parlare di un nuovo computer solo dopo la sua effettiva commercializzazione) presentiamo il nuovo Amiga 3000 che dovrebbe esser presente al prossimo salone dello SMAU.

La decisione di parlarne, più che altro, è dovuta al desiderio di sfatare la "leggenda" di un nuovo modello Amiga che rischierebbe di rendere obsoleto il modello 500.

L'Amiga 3000, infatti, è un computer che, pur "derivato" dal noto a_500 / 2000, è destinato ad un'utenza decisamente professionale. Le sue caratteristiche, all'avanguardia, ed il suo prezzo (si vocifera, per il modello base, di un prezzo che supera i 5 milioni...), lo escludono, di fatto, dall'utenza amatoriale.

A questa, pertanto, rimane la fascia del modello A-500 (o al massimo il 2000).

Il modello A-3000, pur essendo altamente sofi-

Le caratteristiche

Versione 16
Microprocessore Motorola 68030 (**16** Mhz) affiancato da co-processore in virgola mobile (FPU: Floating Point Unit) tipo 68881.

Versione 25
Microprocessore Motorola 68030 (**25** Mhz) affiancato da FPU tipo 68882.

Memoria
Ram Chip da 1 megabyte (espandibile a 2 Mb su scheda).
Ram Fast da 1 Mb, espandibile a 4 Mb su scheda (a 16 Mb con IC Ram da 4 Mbit).

Gestione video
Ad alta risoluzione, dotata di circuiteria che elimina il tipico sfarfallio.

Unità a disco fisso
SCSI interna ad alta velocità (19 millisecondi).

Orologio interno
In tempo reale con batteria tampone.

Slot di espansione
4 slot di espansione Zorro III
2 slot di espansione PC/AT
1 slot di espansione video
1 slot di espansione per memoria Cache / CPU

Connettori in dotazione
Video Amiga (23 pin, 15.75 khz)
Video tipo VGA (15 pin, 31.5 khz)
Audio stereo (canale sin. e des.)
SCSI esterna (25 pin)
SCSI interna (50 pin)
Porta seriale Rs-232
Porta parallela Centronics
Mouse / joy/ penna ottica.

Bus di accesso ai dati
Alla Ram su scheda madre (32 bit)
Alla chip Ram, da parte della CPU (32 bit)
Alla Fast Ram, con controllore custom (32 bit)

Contenitore
Profilo ribassato che integra un totale di tre unità a disco interne (di cui due accessibli esternamente)

Modelli
Amiga 3000 / 16 / 40
68030 / 68881 a 16 Mhz, Ram chip da 1 Mb, Ram Fast da 1 Mb, disco rigido da 40 Mb con tempo di accesso di 19 millisecondi.

Amiga 3000 / 25 / 40
Come Amiga 3000 / 16 / 40 ma con processori 68030 / 68882 a 25 Mh e disco rigido da 40 Mb.

Amiga 3000 / 25 / 100
Come Amiga 3000 / 25 / 40 ma con disco rigido da 100 Mb.

sticato, sarà molto semplice da usare, grazie soprattutto al nuovo sistema operativo (2.0) ed alla nuova interfaccia utente che consente, anche a chi è digiuno di programmazione, di realizzare personalissimi diagrammi di flusso che richiameranno le applicazioni desiderate.

In pratica sarà possibile indicare al computer, mediante il noto sistema icone e finestre, il "percorso" da seguire per svolgere le applicazioni da impostare: word processor, data base, digitalizzazioni sonore e grafiche, richiami di schermate e di souni precedentemenet realizzati da altri paccheti e così via.

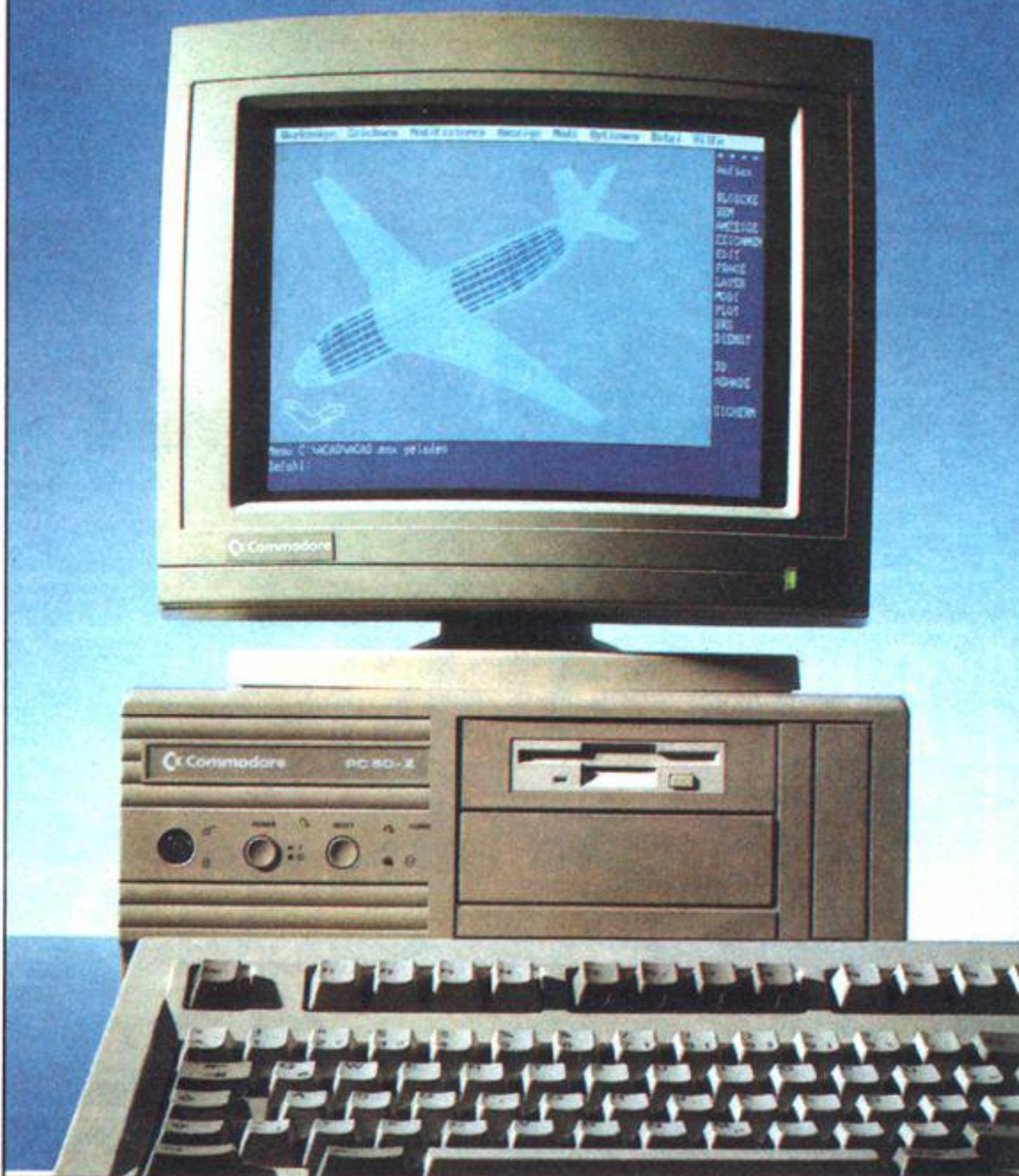
La possibilità, offerta dal nuovo sistema, di creare complesse animazioni senza conoscere nulla di informatica e di scambio di dati tra pacchetti di diversa natura, rende il modello A-3000 ideale per quei campi professionali che operano prevelentemente nel campo della elaborazione grafica e sonora: studi televisivi ed agenzie pubblicitarie; elaborazioni professionali post-video; software di "effetto" e di colloquio, a disposizione del pubblico, da installare in uffici, musei, negozi.

La presenza di due slot per integrare l'ambiente Ms-Dos consente di adoperare il sistema Amiga anche nei luoghi di lavoro prevalentemente dedicati al mondo Ms-Dos.

Naturalmente non ha senso adoperare un A-3000 per limitarsi a lavorare nel mondo Microsoft; la possibilità di elaborare files provenienti da tale ambiente, però, giustifica la presenza degli specifici connettori (immagini provenienti da DTP, data base, testi di word processing e cosi via).

Inutile dire che straordinaria velocità di elaborazione, ed il supporto offerto dai processori matematici, rende A-3000 estremamente concorrenziale nel campo delle elaborazioni grafiche e sonore grazie al prezzo, relativamente basso, di un sistema "ciavi in mano". Dovrebbe dare filo da torcere ache al McIntosh. Staremo a vedere...

Se Amiga non piace...

Chi, finalmente, si è reso conto che il C/64 inizia ad andare un po' stretto per le sempre crescenti esigenze (pur se amatoriali) spesso non sa decidersi verso un sistema o un altro (leggi: Amiga oppure Ms-Dos) a causa della impossibilità di comparare tra loro macchine di caratteristiche (e destinazioni) profondamente diverse tra loro. L'Amiga, infatti, privilegia la parte grafica e sonora (sacrificando, forse, una veloce gestione dei dati su disco); il mondo Ms-Dos, invece, è nato con l'immagine di sistema *professionale* e non riesce a scrollarsi di dosso questa presunta seriosità che è spesso messa in discussione dalla presenza di giochi tanto complessi quanto spettacolari. Il basso prezzo del modello A-500, che imponeva, in precedenza, l'orientamento dei giovani verso questa macchina, oggi è presente anche nei listini degli IBM compatibili. La scelta, pertanto, si è fatta più difficile dal momento che, per fortuna dell'utente finale, il prezzo non rappresenta più una scelta obbligata. Per la gioia dei vostri occhi, quindi, ecco alcune immagini di computer Ms-Dos; di marca, inutile dirlo, rigorosamente Commodore.

A-500 Desk Top Video

Per gli appassionati delle riprese video, che desiderano elaborare i filmati realizzati con le telecamere (operazione che, in gergo, viene denominata **Post - Produzione**) ricordiamo che è disponibile, perfino per il modello Amiga 500, un accessorio Hard - Soft (Amiga Desk Top Video) offerto dalla stessa Commodore.

Si tratta di una procedura che, gestendo la digitalizzazione di immagini provenienti da una telecamera, consente di ottenere numerosissimi effetti, alcuni dei quali sono visibili in queste stesse pagine.

Diversi connettori sono incaricati di prelevare, ed inviare, il segnale video; alcune regolazioni possono essere effettuate anche grazie alle manopole presenti sull'apparecchio.

Questo misura soltanto 290 x 185 x 55 millimetri e pesa poco più di un chilo.

CAMPUS

AMIGA

66 UNA FINESTRA SUL DISCO

Amiga vizia i suoi utenti che, grazie ad icone e finestre, facilita il "colloquio" con le complesse procedure della macchina. In Basic, spesso, è un vero e proprio dramma rintracciare un file nascosto in una directory. La procedura descritta (rigorosamente in AmigaBasic) consente di realizzare una finestra attraverso cui, con semplici comandi, indicare path e file desiderati. Utile da convertire in subroutine da usare in propri programmi.

71 DORARGE E LA CORNACCHIA

Ecco un videogame che vi terrà impegnati sia per digitarlo, sia per giocare (si tratta di catturare una cornacchia con una rete). Ma non crediate di ammirare effetti di anumazione spettacolari: tutt'altro! La cornacchia è formata da tre sprite rettangolari, ed altrettanto banale, graficamente parlando, è la "rete" che dovrete azionare. Nulla a che vedere, insomma, con i veri videogames cui siete abituati. Lo scopo dell'articolo è solo quello di imparare a gestire sprite in Assembly.

78 VIDEO EFFETTI

Alcuni effetti grafici sembra che siano realizzabili solo ricorrendo a linguaggi evoluti. Il programma pubblicato dimostra, al contrario, che qualcosa di simpatico si può realizzare anche in AmigaBasic, che riesce a gestire le librerie della macchina. Un'utile chiacchierata per scoprire le potenzialità notevoli del potente interprete di AmigaRiassunto

Nessun albero viene abbattuto per gli inserti di *Commodore Computer Club*, stampati su carta riciclata al 100%

UNA FINESTRA SUL DISCO

Come realizzare un comodissimo System Requester per scorrazzare in lungo e in largo tra files e directory

di Davide Pagliara

Una finestra di Input per file e directory risulta utilissima nei programmi che richiedono frequenti accessi al disco

Una delle caratteristiche più interessanti ed innovative di Amiga è certamente la possibilità, offerta all'utente, di scorrazzare per lo schermo servendosi del mouse e di fare praticamente di tutto, senza mai toccare la tastiera. Veramente utili sono quelle particolari finestre, presenti in moltissimi programmi, aperte quando si accede al disco per leggerne la directory e per sceglierne un file; queste, di solito, presentano all'interno una lista di files che è possibile far scorrere su e giù clickando su apposite freccine o facendo scorrere il cursore di un Gadget proporzionale.

Esistono molte routines del sistema operativo che facilitano il compito dei programmatori nella creazione del comodo sistema di comunicazione con Amiga; il loro uso risulta abbastanza agevole se si programma in Assembler o in C, un po' meno se si programma in Basic. Il programma presentato, scritto interamente in AmigaBasic, viene in aiuto a tutti coloro che vorrebbero utilizzare una finestra del tipo descritto precedentemente al fine di leggere le directory ed i files presenti su disco ed, eventualmente, di selezionarne il nome.

C'e' file e files

Amigabasic dispone del comando **Files** che serve, appunto, a listare le directory di un disco. Purtroppo non è molto utile, se richiamato da programma, in quanto si limita a scaricare l'intera lista dei files e delle directory sulla finestra attualmente attiva, senza dare la possibilità di passare da una directory all'altra o di selezionare il nome di un file. Per aggirare l'ostacolo, è necessario ricorrere ad alcune routines della **dos. library** che permettono di accedere al disco e di trarne informazioni molto utili.

L'accesso ad una directory o ad un file è reso possibile tramite l'utilizzo del **FIB** (**F**ile **I**nfo **B**lock) mentre per accedere al FIB è necessario un **Lock** (chiave) riferito alla directory che si vuole esaminare. La funzione Lock della *dos.library* viene in aiuto; se volessimo Lock-are una directory dovremmo scrivere...

dl& = Lock& (SADD (unit$), n)

...in cui **dl&** è il lock cercato, **unit$** contiene il pathname della directory o del file terminato con *Chr$ (0)*, **n** è il modo di accesso e può valere **-1** (**Exclusive_lock**) oppure **-2** (**Shared_lock**). Amiga è un sistema multitasking e, come tale, deve condividere le risorse con altri tasks; con **-2** si offre la possibilità ad altri tasks di leggere; con **-1**, invece, no.

Esempio. Per Lock-are la directory **Devs:** del disco presente nel drive **Df0:** in maniera condivisa, dovremmo scrivere...

unit$ = "df0: Devs" + CHR$ (0)
dl& = Lock& (SADD (unit$), -2)

Una volta Lock-ata la directory, possiamo procedere a riempire la struttura FileInfoBlock utilizzando la funzione **Examine ()** la cui sintassi è:

Examine (dl&, l&)

...in cui **dl&** è il lock ottenuto precedentemente e **l&** è l'indirizzo della zona di memoria che ospiterà la struttura **FIB**; tale indirizzo pur essere ottenuto con...
l& = Allocmem& (260, 65537&)
A questo punto possiamo leggere la struttura FIB e ricavare i dati relativi alla directory o al file selezionato. La struttura File Info Block è cosl organizzata:

FileInfoBlock:
LONG fib_DiskKey
LONG fib_DirEntryType
char fib_FileName
LONG fib_Protection
LONG fib_EntryType
LONG fib_Size
LONG fib_NumBlocks
struct fib_DateStamp
char fib_Comment

Date Stamp, a sua volta, è un'altra struttura cosl organizzata:

DateStamp:
LONG ds_Days; Giorni trascorsi dal 1. 1. 1978
LONG ds_Minute; Minuti dalle ore 00: 00
LONG ds_Tick; Tick nel minuto corrente (1 tick = 1/50 sec.)

Da Basic si avrà:

Peek (l& + 8) Nome del file o della directory (max 108 caratteri) terminato da Chr$ (0).
Peekl (l& + 116) Maschera di protez. (rwed)
Peekl (l& + 120) Tipo: file = 0; directory > 0
Peekl (l& + 124) Lunghezza in byte
Peekl (l& + 128) Lunghezza in blocchi
Peekl (l& + 132) Indirizzo della struttura DateStamp
Peekl (l& + 144) Commento (max 116 caratteri) terminato da CHR$ (0).

La Directory

Per ottenere l'elenco dei files presenti in una directory bisognerà agire in maniera leggermente diversa: dopo aver Lock-ato la directory interessata e chiamato la routine Examine, bisognerà utilizzare ripetutamente **ExNext()**, un'altra routine della dos. library. La sintassi di ExNext () è la seguente:
b& = ExNext (dl&, l&);
l&, questa volta, dovrà essere l'indirizzo di una zona di memoria, allocata con Allocmem (), abbastanza lunga da ospitare il nome di tutti i files presenti nella directory. La variabile **b&** vale **-1** se non ci sono stati errori. La routine ExNext dovrà essere chiamata ripetutamente fino all'esaurimento dei files presenti nella directory selezionata cioè fino a quando, chiamando la funzione **IoErr ()**, non otterremo il valore **232** (Error_no_more_entries). Esempio:

```
l& = Allocmem (1000, 65537&)
unit$ = "df0: devs" + CHR$ (0)
dl& = Lock& (SADD (unit$), -2)
CALL Examine (dl&, l&)
doserr = IoErr (0)
WHILE doserr < > 232
CALL ExNext (dl&, l&): REM Attenzione
......qui utilizziamo la stessa area
......di memoria utilizzata prima con
......Examine ().
WEND
```

Per maggiori dettagli fare riferimento al listato **Open File** pubblicato.

Il programma

Il programma presentato non è altro che una subroutine, da collocare in coda ai propri programmi, da richiamare ogni volta che si vuole accedere alla lista dei files presenti sul disco. Una volta lanciato, esso apre una finestra all'interno della quale appare subito la lista dei files presenti sul disco posto in **df0:** ; è possibile far scorrere su e giù la lista clickando sulle freccine poste a destra della stessa. Tenendo premuto il tasto sinistro del mouse su una delle frecce,

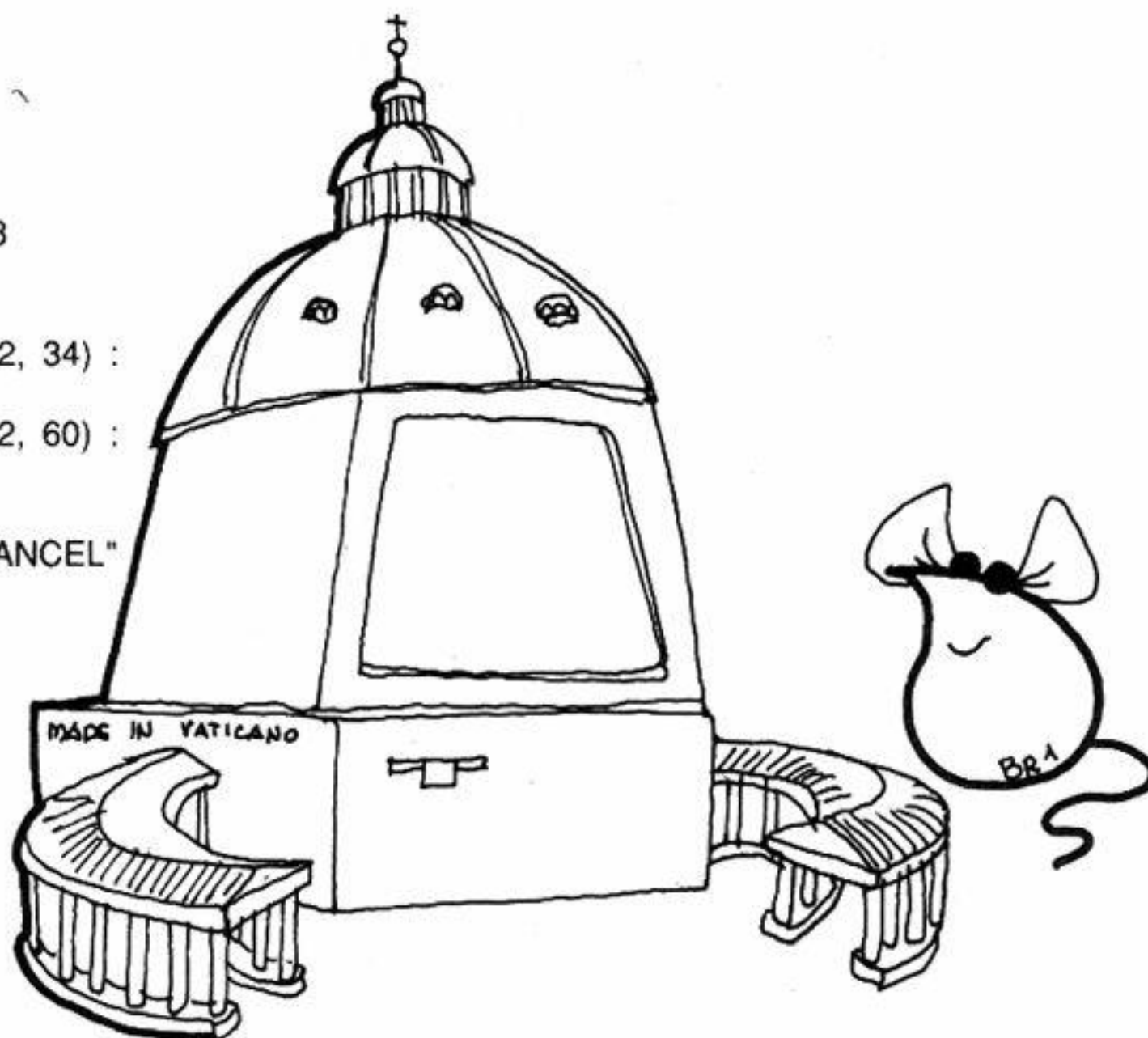
VIETATO AI MINORI

E' probabile che i neo-utenti di computer non riescano a ben comprendere l'utilità della procedura software descritta nelle presenti pagine. Questa è infatti destinata a coloro che sono già padroni delle principali tecniche di programmazione e desiderano approfondire il modo in cui un elaboratore organizza i vari dati all'interno della memoria.
Il programma, pertanto, rappresenta un invito ai lettori più esperti, soprattutto a coloro che, considerando il listato come una base di partenza, riescano a pervenire a procedure più interessanti e capaci, magari, di offrire applicazioni di più ampio respiro.
Se hai incominciato da poco, comunque, non scoraggiarti! Tutti coloro che, oggi, vantano una particolare competenza nel campo dell'informatica hanno iniziato con un banale Print "Pippo". Perchè non dovresti riuscire anche tu?...

Se dovessero verificarsi errori "estetici" (di incolonnamen to) provate ad inserire spazi bianchi nelle stringhe associate ai comandi Print

```
REM --- Esempio ---
PRINT "SELEZIONARE UN FILE."
GOSUB OPENLIBRARY 'apre le librerie
loop:
CALL OPENFILE 'chiamata ad openfile
CLS: WINDOW OUTPUT 1
PRINT "E' STATO SELEZIONATO IL DRAWER: "; dra-
wer$
PRINT
PRINT "FILE: "; file$
PRINT
INPUT "Vuoi selezionare un altro file (S/N)"; d$
IF d$ = "s" THEN loop
GOSUB CLOSELIBRARY
END
REM --- fine esempio ---
REM Open-File
REM 1990
REM by Davide Pagliara
REM Massafra (Ta)
REM *** INIZIO SUBROUTINE OPENFILE ***
REM *** Attenzione: LA VARIABILE l& E' RISERVATA ***
SUB OPENFILE STATIC
SHARED drawer$, file$, l&
undofile$ = file$: undob$ = drawer$
sp$ = "                                     " '37 spazi
GOSUB Inmem
GOSUB Openwindow
GOSUB drawer1
WHILE MOUSE (0) < > 0 : WEND
loop.mouse:
IF MOUSE (0) = 0 THEN
 IF qw > 150 THEN qw = 1
 x = MOUSE (1) : y = MOUSE (2)
 GOTO loop.mouse
END IF
es:
qw = qw + 1
IF qw > 150 THEN start
IF MOUSE (0) = 0 THEN qw = 1: GOTO start
GOTO es
start:
IF x > 260 AND x < 284 THEN
 IF y > 22 AND y < 46 THEN GOSUB fs
 IF y > 48 AND y < 72 THEN GOSUB fg
END IF
IF y > 98 AND y < 118 THEN
 IF x > 18 AND x < 78 THEN uscita
 IF x > 120 AND x < 180 THEN GOSUB parent
 IF x > 224 AND x < 284 THEN
 drawer$ = undob$: file$ = undofile$
 GOTO uscita
 END IF
END IF
IF y > 6 AND y < 16 THEN GOSUB drawer
IF y > 78 AND y < 89 THEN GOSUB file
IF y > 24 AND y < 72 AND x < 256 THEN GOSUB lista
GOTO loop.mouse
parent: 'calcola il nome della directory genitrice
c$ = RIGHT$ (drawer$, 1)
WHILE (c$ < > "/" AND c$ < > ":" AND LEN (drawer$) > 1)

 drawer$ = LEFT$ (drawer$, LEN (drawer$) -1)
 c$ = RIGHT$ (drawer$, 1)
WEND
IF c$ < > ":" AND drawer$ < > "" THEN
 drawer$ = LEFT$ (drawer$, LEN (drawer$) -1)
END IF
LOCATE 2, 1: PRINT sp$
LOCATE 2, 1: PRINT RIGHT$ (drawer$, 37)
GOSUB drawer1
RETURN
lista: 'calcola file/directory selezionato
t = INT (y/8) -3
IF scr > 6 THEN t = t + scr-6
IF LEFT$ (nfile$ (t), 5) = "(DIR)" THEN
 c$ = ""
 IF RIGHT$ (drawer$, 1) < > ":" AND drawer$ < > "" THEN
c$ = "/"
 drawer$ = drawer$ + c$ + MID$ (nfile$ (t), 7)
 LOCATE 2, 1: PRINT sp$
 LOCATE 2, 1: PRINT RIGHT$ (drawer$, 37)
 GOSUB drawer1
 file$ = "": RETURN
END IF
file$ = nfile$ (t)
IF drawer$ = "" THEN file$ = ""
GOSUB file1
RETURN
fs: 'Scrolling della lista dei files
LOCATE 9, 1
IF scr < file THEN
 SCROLL (8, 24) - (256, 71), 0, -8
 PRINT " "; LEFT$ (nfile$ (scr), 31)
 scr = scr + 1
END IF
RETURN
fg:
LOCATE 4, 1
IF scr-6 > 0 THEN
 SCROLL (8, 24) - (256, 71), 0, 8
 PRINT " "; LEFT$ (nfile$ (scr-7), 31)
 scr = scr-1
END IF
RETURN
drawer: 'digitazione della directory da leggere
LOCATE 2, 1: PRINT sp$
LOCATE 2, 1
INPUT "", drawer$
LOCATE 2, 1: PRINT RIGHT$ (drawer$, 37)
drawer1:
GOSUB ldir
IF er = 0 THEN
 LOCATE 4, 1
 FOR t = 0 TO 5: PRINT LEFT$ (sp$, 31) : NEXT
 LOCATE 4, 1: scr = file
 IF file > 6 THEN scr = 6
 FOR t = 0 TO scr-1
```

```
 PRINT " "; LEFT$ (nfile$ (t), 31) : NEXT
END IF
RETURN
file: ' digitazione del file selezionato
LOCATE 11, 1: PRINT sp$
LOCATE 11, 1: INPUT "", file$
file1:
LOCATE 11, 1: PRINT sp$
LOCATE 11, 1: PRINT LEFT$ (file$, 37)
RETURN
ldir: 'lettura della directory
file$ = "": GOSUB file1
IF drawer$ = "" THEN RETURN
er = 0: unit$ = drawer$ + CHR$ (0)
dl& = Lock& (SADD (unit$), -2)
IF dl& = 0 THEN CALL UnLock (dl&) : er = 1: RETURN
FOR t = 1 TO 100 : nfile$ (t) = "": NEXT
CALL Examine (dl&, l&)
file = 0
doserr = IoErr& (0)
WHILE doserr < > 232 AND file < 100
 CALL ExNext (dl&, l&)
 doserr = IoErr& (0)
 nfile$ (file) = ""
 cm& = l& + 8
 p = PEEK (cm&)
 WHILE p
 nfile$ (file) = nfile$ (file) + CHR$ (p)
 cm& = cm& + 1
 p = PEEK (cm&)
 WEND
 fd = PEEKL (l& + 120)
 IF fd > 0 THEN nfile$ (file) = "(DIR)" + nfile$ (file)
 file = file + 1
WEND
file = file-1: nfile$ (file) = ""
RETURN
Openwindow: ' finestra e grafica
WINDOW 15, "Open File", (10, 20) - (305, 140), 18
COLOR 1
LOCATE 2, 1: PRINT drawer$
AREA (272, 24) : AREA (262, 34) : AREA (282, 34) :
AREAFILL
AREA (272, 70) : AREA (262, 60) : AREA (282, 60) :
AREAFILL
COLOR 3
LOCATE 14, 3: PRINT "  OK        PARENT      CANCEL"
LINE (0, 16) - (294, 16), 3
LINE (0, 6) - (294, 6), 3
LINE (0, 89) - (294, 89), 3
LINE (0, 78) - (294, 78), 3
LINE (18, 98) - (78, 118), 1, b
LINE (120, 98) - (180, 118), 1, b
LINE (224, 98) - (284, 118), 1, b
LINE (259, 48) - (285, 72), 1, b
LINE (259, 22) - (285, 46), 1, b
LINE (269, 34) - (275, 44), 1, bf
LINE (269, 50) - (275, 60), 1, bf
LOCATE 1, 16: PRINT " Drawer "
LOCATE 10, 17: PRINT " File "
COLOR 1
RETURN
lnmem: 'alloca la memoria necessaria
IF l& = 0 THEN
 l& = Allocmem& (1000, 65537&)
 IF l& = 0 THEN
 LOCATE 4, 1
 PRINT "MEMORIA INSUFFICIENTE"
 FOR t = 1 TO 12000: NEXT: GOTO uscita
 END IF
 DIM nfile$ (100)
 drawer$ = "df0:"
END IF
RETURN
uscita:
WINDOW CLOSE 15
END SUB
OPENLIBRARY: 'apre le librerie
DECLARE FUNCTION IoErr& LIBRARY
DECLARE FUNCTION Lock& LIBRARY
DECLARE FUNCTION Allocmem& LIBRARY
LIBRARY "dos.library"
LIBRARY "exec.library"
RETURN
CLOSELIBRARY: 'chiude le librerie
CALL FreeMem (l&, 1000)
LIBRARY CLOSE
RETURN
```

Preparate con cura il dischetto di boot per evitare errori del tipo "file not found"

si ottiene lo scorrimento veloce della lista. Clickando sul nome di uno dei files visualizzati, se ne ottiene la selezione nella riga denominata **File**; se invece si seleziona una directory, essa verrà letta ed i suoi files visualizzati nella lista. Portando il puntatore nella riga denominata **Drawer** o nella riga File e premendo il tasto sinistro del mouse, è possibile inserire, da tastiera, il nome della directory (Drawer) o del file da selezionare; al termine dell'immissione premere Return. In basso vengono visualizzati tre rettangoli denominati:

OK - Questo tasto chiude la finestra Open File e restituisce nelle variabili, rispettivamente, drawer$ e file$ il Pathname della directory ed il nome del file selezionati.

Parent - Legge la directory genitrice. A questo proposito è bene notare che esiste una funzione della dos.library chiamata **ParentDir()** che richiede, come argomento, il lock associato alla directory attuale e che restituisce il lock della directory genitrice. Nel programma, tuttavia, non si fa uso di tale funzione.

Cancel - Chiude la finestra Open File presentando, in drawer$ e file$, il Pathname e il nome del file selezionato prima dell'ultima chiamata ad Open File, indipendentemente dal drawer e dal file selezionati attualmente.

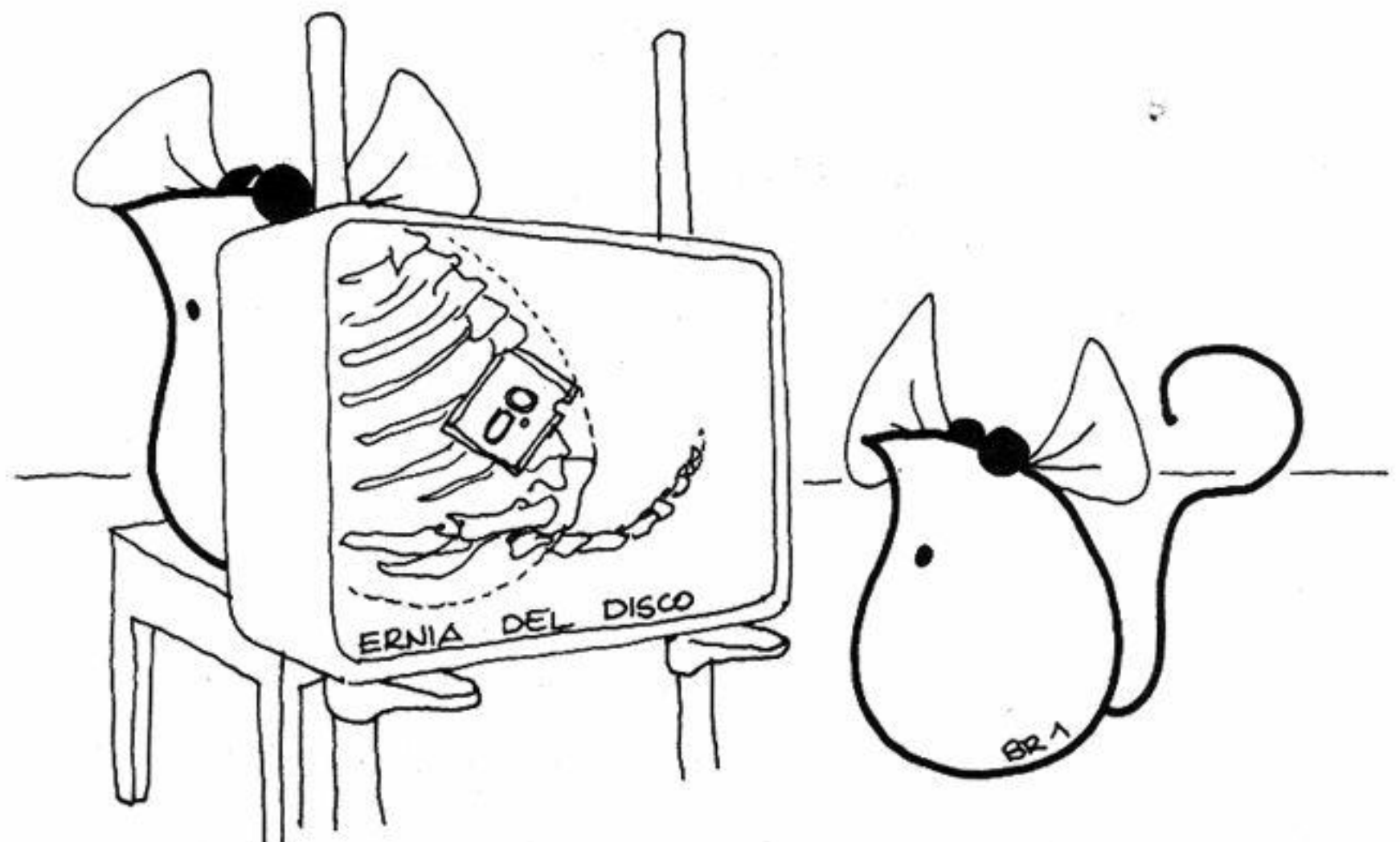
Osservazioni

Il programma Open File deve essere collocato in coda ai vostri programmi; per far ciò conviene salvarlo in codice Ascii mediante il comando...

SAVE "df0: Open File", a

...e, in seguito, *fonderlo* ai vosri programmi per mezzo del comando *Merge*. Per il corretto funzionamento di Open File è necessario chiamare le subroutines **Openlibrary** e **Closelibrary** rispettivamente all'inizio ed alla fine del vostro programma; Open File, invece, si attiva con **Call Openfile.** Si consiglia, comunque, di guardare l'esempio proposto insieme al listato di Open File. La finestra di Open File si adatta benissimo ad ogni schermo avente qualsiasi risoluzione; per fare ciò, tuttavia, bisognerà specificare nel comando Window presente nella subroutine OpenWindow, l'ID dello schermo al quale si vuole fare riferimento. Confrontare, per questo, il manuale dell'AmigaBasic.

Attenzione: la variabile **I&** dovrà essere considerata *riservata* e quindi non dovrà mai comparire nei programmi Basic che utilizzino Open File. **Ricordate che sul disco da cui fate il boot devono essere presenti i file dos. bmap ed exec. bmap; li trovate già pronti sul disco Extras, directory BasicDemos.**

Al fine di un utilizzo più confortevole, il programma Open File avrebbe potuto servirsi di Gadgets di tipo stringa per la immissione da tastiera dei nomi delle directory o dei files nonchè di un Gadget proporzionale per lo scorrimento della lista su schermo; tuttavia, data la difficoltà di trattare, da Basic, le strutture relative ai Gadgets, che, peraltro, avrebbero allungato ulteriormente il listato, si è optato per una risoluzione più spartana facendo ricorso al classico comando Input del Basic.

DORARGE E LA CORNACCHIA

Un programma, da digitare con facilità, ci consente di esaminare un modo di gestire la grafica in Assembly

di Donato De Luca

Recentemente è apparso, su un noto quotidiano italiano, un articoletto in cui si narrava di una cornacchia impazzita che attaccava i passanti per la strada.

Nel programmino proposto in questo articolo voi impersonate **Dorarge**, l'nviato di un'associazione ecologica per la protezione degli uccelli. Il vostro compito è quello di catturare la cornacchia malefica servendovi di una rete. Tuttavia, manco a farlo apposta, ogni volta che catturate una cornacchia ne spunta fuori un'altra.

Scherzi a parte il semplicissimo giochino mostra la gestione degli **Sprites Hardware** (modo automatico **DMA**) in assembler, la creazione di un programma per il **Copper**, la creazione e la gestione di un **Playfield**, ed altre cose. Come i più avranno certamente capito, non è possibile trattare in maniera approfondita tutti gli argomenti sopra citati; basti pensare che l'**Amiga HardWare Reference Manual**, (uno dei quattro *vangeli* dei programmatori Amiga, famigerato per la scarsa chiarezza) dedica circa 100 pagine ad una parte degli argomenti citati.

Ci limiteremo, quindi, solo ad una panoramica sulla programmazione del 68000 in ambiente Amiga. Ora daremo uno sguardo (ma solo uno) al Copper, agli Sprites HardWare, ed al display.

❑❒

Il ruolo del Copper

Il Copper si occupa di fare tante belle cosette, tra cui il ripristino dei **puntatori dinamici**, cioè quei puntatori che vengono alterati durante la generazione del Display (puntatori ai BitPlanes, alle strutture sprite DMA, se utilizzate ecc.). Il Copper, inoltre, può fare in modo che venga generata una richiesta d'**interrupt Hardware**. Altro compito normalmente svolto dal Copper è la scrittura nei **registri colore**, in modo da utilizzare diversi valori per il medesimo registro colore nel corso della generazione del Display. Il Copper può essere usato per attivare il **Blitter** (tuttavia, per far ciò, si deve settare a 1 il bit n.1 del registro Copdan).

Grazie al Copper, infine, disporre di 1000 Sprites Hardware, in un unico quadro, è abbastanza semplice.

❑❒

Sprite

L'Amiga possiede 8 graziosi Sprites Hardware, larghi 16 pixels ciascuno e alti a piacere. Normalmente ogni Sprite ha 4 colori (3 + trasparenza), poichè per raffigurare uno Sprite si usano due bit per pixel. In questo modo è facile rendersi conto che per ogni pixel si possono avere 4 combinazioni (2 exp 2 = 4).

Tuttavia i progettisti di Amiga hanno pensato bene di offrire anche la possibilità di gestire sprites a 16 colori (15 + trasparenza). Per operare il sortilegio si è pensato di *fondere* gli sprite a coppie, in modo che ogni pixel di uno sprite sia rappresentato con 4 bit (2 exp 4 = 16) anzichè due. Ovviamente il numero totale di sprites scende da 8 a 4. E' da notare che le quattro coppie sono **indipendenti** ed è quindi possibiule, ad esempio, avere uno sprite a 16 colori e gli altri a 4. Esistono due sistemi per gestire gli sprites: quello automatico e quello manuale.

Quest'ultimo (quasi mai usato) consiste nel far passare i dati degli sprites al circuito che li genera direttamente dal boss (68000, 68010, 68020, 68030 o quello che avete..). Tuttavia anche se il vostro Boss và a 50 Mhz questo metodo fa sprecare tempo inutilmente.

L'altro sistema, quello automatico, fa uso di 8 canali DMA per inviare i dati a destinazione. Per far ciò, prima di tutto si devono abilitare i canali DMA relativi agli sprites, successivamente si deve far puntare ogni canale ad una strut-

Un minigioco in Assembly, ma non aspettatevi grandi cose...

Il Copper

Il Copper è un coprocessore, sincronizzato con il pennello elettronico, il cui set di istruzioni è formato dalle sole istruzioni

Move, Wait, Skip

Move: trascrive in un indirizzo, a partire da $dff000, un valore a 16 bit

Nota: a partire da $dff000 vi sono i **registri** dei vari chip custom di Amiga: alcuni di essi sono di **sola scrittura** (Write Only) ed il tentativo di accesso in lettura, da parte del 68000, non è molto salutare.

Registri Write Only sono i puntatori ai bitplanes, i registri colore ecc. Vi sono anche registri di **sola lettura** (Read Only), tipo Joy1Dat, Joy0Data, Clxdat, Potgor ecc. E' da notare che tutti i registri occupano 16 bit, anche gli Strobe. Ad alcuni registri il Copper non può accedere mentre ad altri può accedere solo se abilitato dal programmatore.

Wait: attende che il pennello elettronico sia giunto in una determinata posizione prima di elaborare la successiva istruzione del programma.

Skip: passa alla prossima istruzione del programma del Copper, se, e solo se, il pennello è arrivato in una certa posizione.

Nota: per terminare un programma del copper si usa **$ffff, $fffe** il cui significato è: attendi fino alla posizione 254 della linea 255.

Tale posizione non verrà mai raggiunta perchè, quando si arriverà alla fine della generazione del Display (cioè quando inizierà il Vertical Blank) l'istruzione non risulterà mai completata. Si deve infatti tener presente che durante ogni Vertical-Blank il P.C. (program counter) del Copper viene fatto puntare nuovamente all'inizio del programma. Di conseguenza qualunque istruzione si trovasse dopo $ffff, $fffe non sarà mai eseguita.

Il programma elaborato dal Copper viene chiamato **CopperList** e deve risiedere tassativamente nella ChipRam.

tura dati (che chiameremo Struttura Sprite DMA) le cui prime due words sono dette words di controllo, poichè non contengono dati della sagoma, ma contengono nei primi tre bytes altri dati di secondaria importanza (la posizione di start e stop verticale e quella di start orizzontale). Nel quarto byte sono contenuti dati relativi alla modalità dello sprite.

La fine di una struttura sprite DMA viene indicata con due words nulle. In questo modo si ferma il canale DMA; se però, al posto delle due words nulle, inseriamo altre due words di controllo, otterremo due sprites di altro tipo, che chiameremo **sprite virtuale**.

Il Display

Sull'Amiga il Display è composto dall'insieme dei **PlayFields** e dagli sprites. I playfields sono composti da un massimo di 6 bitplanes. Per creare un display *decente* si deve fissare il modo grafico scegliendo tra i 4 possibili (a meno che non avete già l'ECS, in tal caso la scelta non si impone..) e cioè 320 x 200, 640 x 200, 320 x 400, 640 x 400. Ovviamente questi sono i valori per il sistema NTSC, ma noi che mangiamo PAL abbiamo 255 linee in modo non interlace.

Si deve quindi fissare il **modulo** adeguato (cioè la quantità da sommare all'inizio di una linea per passare alla successiva), il punto di **start** della finestra video, quello di **stop**, il numero di **bitplanes** utilizzati, lo **scroll** (meglio dire shift), i valori di **Ddfstart** e **Ddfstop** e i valori dei **colori** desiderati nei registri colore utilizzati.

VIETATO AI MINORI

E' probabile che i neo-utenti di computer non riescano a ben comprendere l'utilità della procedura software descritta nelle presenti pagine. Questa è infatti destinata a coloro che sono già padroni delle principali tecniche di programmazione e desiderano approfondire il modo in cui un elaboratore organizza i vari dati all'interno della memoria.

Il programma, pertanto, rappresenta un invito ai lettori più esperti, soprattutto a coloro che, considerando il listato come una base di partenza, riescano a pervenire a procedure più interessanti e capaci, magari, di offrire applicazioni di più ampio respiro.

Se hai incominciato da poco, comunque, non scoraggiarti! Tutti coloro che, oggi, vantano una particolare competenza nel campo dell'informatica hanno iniziato con un banale Print "Pippo".

Perchè non dovresti riuscire anche tu?...

Per chi inizia

L'assemblatore usato è il **Seka**. Sappiamo benissimo che esistono in commercio decine di assemblatori migliori (noi stessi usiamo spesso il **DevPac**); tuttavia ricordiamo che la serie di articoli sull'Assembly è destinata a lettori per lo più principianti, per i quali è relativamente facile entrare in possesso di una versione del Seka.

Inoltre il Seka è molto facile da utilizzare. Chi non volesse usare il Seka, tuttavia, non deve fare altro che inserire qualche piccola modifica per assemblare i programmi di volta in volta proposti.

Per Assemblare (Seka2.12)
Caricate il Seka; alla richiesta Chip o Fast scrivete **C**; Alla seconda domanda scrivete il numero dei k desiderati (**20** sono più che sufficienti per i programmi di solito proposti). A questo punto potete scrivere il listato di queste pagine tramite l'editor incorporato oppure tramite un Editor esterno. Se avete scelto di usare un Editor esterno ricordatevi di aggiungere **.S** al nome del file. Per caricarlo dovrete scrivere **R** e, alla richiesta del nome, digitate il nome del file. Una volta che il file è presente in memoria lo assemblerete digitando **A**, ed ignorando la richiesta di opzioni. Se vi sono state segnalazioni di errore dovrete effettuare le correzioni del caso. E' da notare che il Seka interrompe le operazioni ogni volta che incontra un errore. Salvate il sorgente con **W**, rispondendo alla richiesta del nome con quello che vi pare. State attenti che il Seka non controlla se vi era già un file esistente con il nome scelto quindi attenti alle sovrascritture! Una volta salvato il file fate partire il programma con **G**. Se non volete mettere un BreakPoint ignorate la richiesta **BreakPoint?** Se tutto è andato bene avrete ottenuto un file eseguibile direttamente da Cli.

Per far ciò scrivete **Wo**, ed alla richiesta Mode scrivete **C** (in questo modo dite all'AmigaDos di caricare il codice in Chip). Ovviamente fatelo solo se avete bisogno che il programma sia allocato in Chip! Se invece non interessa la zona in cui Amigados allocherà il programma, ignorate la richiesta **mode**.

Per "uscire" da Seka digitate il punto esclamativo (**!**).

La sintassi Seka non è molto diversa da quella necessaria per usare altri assemblatori per Amiga

Il listato

Il programmino allegato è un esempio pratico degli argomenti toccati. Lo scopo del gioco è di colpire la cornacchia. Non aspettatevi di vedere una vera cornacchia: sia questa, sia Dorarge, sono rappresentati da blocchetti informi. Nel programma vi sono tre sprites; uno per Dorarge, uno per Corny (la cornacchia) ed uno per la rete.

All'inizio del programma si agisce sullo **stack**; viene quindi prelevato l'indirizzo base **di Exec** e, successivamente, viene aperta la **Graphic Library** per determinare la posizione della Copper List di sistema.

L'indirizzo di questa si trova 50 bytes oltre l'indirizzo della Gfx. Ricavata la preziosa informazione, la salviamo e, al posto dell'indirizzo della CopperList di sistema, inseriamo l'indirizzo della **nostra** "SuperCopperList". Per questa operazione, tuttavia, si deve disattivare il Copper onde evitare pasticci vari. Successivamente si allocano in memoria CHIP le strutture Sprite Dma. Dato che gli Sprites utilizzati sono tre saranno allocate altrettante strutture Sprite Dma. E' bene notare che non viene effettuato un controllo se Exec ha realmente fornito la memoria richiesta. Si chiede quindi la memoria per il Bitplane e si abilita l'interrupt di livello 3 (generato ad ogni VerticalBlank); al posto della normale routine di interrupt viene eseguita la nostra routine.

In **Main** viene testato in continuazione il pulsante sinistro del Mouse: se viene rilevata la pressione il programma termina, restituisce la memoria presa all'inizio, ripristina i valori originari del vettore d'irq di livello 3 e del puntatore alla copper list attuale (50 bytes dopo Gfx), chiude la Gfx e torna da dove è partito.

Passiamo ora ad esaminare velocemente **Nucleo**, il cuore della spietata caccia alla cornacchia. All'inizio, come al solito, salviamo tutto sullo stack e facciamo puntare **A0** alla struttura dati della cornacchia. Successivamente sommiamo all'**ascissa** della cornacchia il **DeltaX** attuale. Poi viene chiamata **Pos**, la quale ha il

COPIALO PER TELEFONO

Anche i listati presenti in queste pagine possono esser tirati giù per mezzo del modem; se, ovviamente, ne possedete uno.
La procedura per collegarsi con la nostra banca dati (attiva 24 ore su 24) è riportata su altra parte della rivista.
Chi non possiede il modem (che aspettate a procurarvelo?) può tuttavia richiedere il dischetto, contenente il software, presso il nostro servizio arretrati.

La gestione degli sprite è uno degli argomenti affrontati in queste pagine

compito di ricalcolare le due words di controllo della struttura Sprite Dma. Viene quindi fatto puntare A0 alla struttura dati della rete. Si controlla se vi è un lancio in corso e, in caso affermativo, si decrementa di 4 l'ordinata della rete. Se non vi è nessun lancio in corso lo sprite della rete vien fatto sparire. In **Next** A0 punta alla struttura di Dorarge.

In **Joy** si controlla se l'asticella è stata mossa e se vi è un lancio in corso. In caso negativo si controlla se è premuto il pulsante di fuoco della porta 2. In caso affermativo viene lanciata la rete, in caso negativo si decrementa di uno il contatore di lancio. Viene quindi chiamata **Pos** il cui compito è di aggiornare, con i dati contenuti nella struttura dati Sprite, le due words di controllo contenute nella struttura Sprite Dma.

TestCollision: in questa routine si controlla se vi è stata una collisione tra lo sprite 0 e lo sprite 2. Se vi è stata, si incrementa di 1 il delta X della cornacchia, e la si posiziona a (0, 0). Se non si sono scontrati, pazienza.

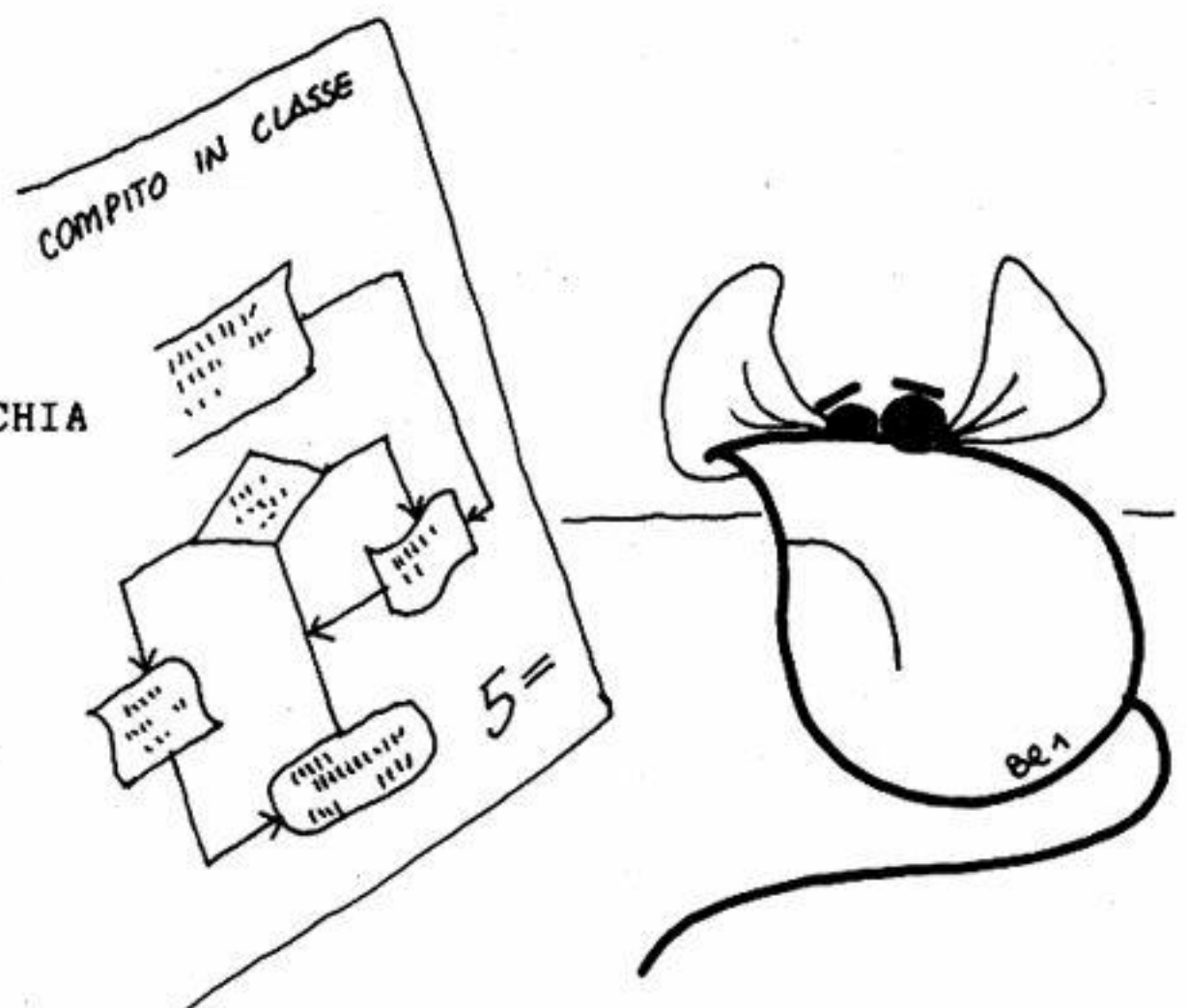
```
; DORARGE E  LA CORNACCHIA
;        written by
;     Donato De Luca

EXECBASE      = $4
ALLOCMEM      = -198
FREEMEM       = -210
OPENLIBRARY   = -408
CLOSELIBRARY  = -414

INTENA = $DFF09A
DMACON = $DFF096
JOY1DAT= $DFF00C
CLXDAT  = $DFF00E
CIAA    = $BFE001

VI3 = $6C

movem.l d0-d7/a0-a6,-(sp);Tutto Stack
move.l EXECBASE,a6         ;ExecBase->a6
lea GFXNAME,a1             ;Apro la
jsr OPENLIBRARY(a6)        ;Graphic
move.l d0,a5               ;In d0,non controllo
move.w #$0080,DMACON       ;Copper bye bye.
move.l 50(a5),OLDCOPPER    ;Salvo ind Sys CList
move.l #COPPERLIST,50(a5);Punto mia CList.
move.w #$8080,DMACON       ;Resuscito Copper.

lea TABELLAMOSTRI,a2       ;Dove prendo SPRITE
move.w #2,d2               ;Quanti ( - 1 )
CHIEDIELEMOSINA:           ;Alloco SPRITE in CHIP
move.l (a2)+,a3            ;Indirizzo SPRITE att.
move.l #$0002,d1           ;VOGLIO LA CHIP!!!!!!
move.l #7*5,d0             ;Se non mi dai 35 bytes
jsr ALLOCMEM(a6)           ;mi metto a piangere
move.l d0,MEMOGG           ;Me li hai dato?Vero??
move.l 12(a3),a0           ;Copper seg sprite att
move.w d0,6(a0)            ;LOW word
swap d0                    ;Trallallero,trallalla'
move.w d0,2(a0)            ;HIGHT word
move.l MEMOGG,4(a3)        ;Salvo ind reale SPR
```

```
move.l 8(a3),a0              ;Modello SPRITE
move.l MEMOGG,a1             ;Dove lo metto
move.l #35,d0                ;Quanti bytes

TRANLOOP:
move.b (a0)+,(a1)+           ;Cercasi belle ragazze
dbra d0,TRANLOOP             ;per modelli SPRITE..
dbra d2,CHIEDIELEMOSINA      ;Tutto per NSPRITE-1

move.l #320*256/8,d0         ;255 = PAL
move.l #$10002,d1            ;LA VOGLIO LINDA!!!!
jsr ALLOCMEM(A6)             ;Sbrigarsi
move.l d0,PIANO1             ;Mi fido
lea BITPLANE1,a0             ;Faccio puntare
move.w d0,6(a0)              ;il puntatore
swap d0                      ;al piano
move.w d0,2(a0)              ;appena preso
move.l PIANO1,a0             ;Salam
add.l #$1e28,A0              ;mSala
move.l #$180c0400,(a0)       ;amSal
move.l #$140a0400,$28(a0);ecc...
move.l #$12090400,$50(a0);
move.l #$12090400,$78(a0);
move.l #$12492410,$a0(a0);
move.l #$14ea7438,$c8(a0);
move.l #$184c2790,$f0(a0);

move.w  #$8020,INTENA        ;Attivo VERTB
move.l VI3,OLDIRQ            ;Salvo vet IRQ3
move.l #NUCLEO,VI3           ;Nucleo-> vet IRQ3


MAIN:                        ;Aspetto finche` il
btst #6,CIAA                 ;Pulsante sin mouse
bne.s MAIN                   ;non viene premuto

EXIT:
move.w #2,d3                 ;Quanti mostri?
lea TABELLAMOSTRI,a2         ;Dove sono le str?
BUCOLICO:                    ;Rido la memoria
move.l (a2)+,a3              ;a EXEC
move.l 4(a3),a1              ;d0=quantita`
move.l #35,d0               ;a1=indirizzo
jsr FREEMEM(a6)              ;
dbra d3,BUCOLICO             ;
move.l #320*256/8,d0         ;Rido il piano
move.l PIANO1,a1             ;
jsr FREEMEM(a6)              ;


;rispristino vettore irq
move.l OLDIRQ,VI3            ;
move.w #$0080,DMACON         ;Re:Bye Bye Copper
move.l OLDCOPPER,50(A5)      ;Punto Old CList
move.w #$8380,DMACON         ;Resuscito Copper.
move.l a5,a1                 ;ind Graphic in A0
jsr CLOSELIBRARY(a6)         ;chiudo la graphic.
movem.l (sp)+,d0-d7/a0-a6;Butto tutto fuori
rts                          ;Torna dal papa.

NUCLEO:
movem.l D0-D7/A0-A6,-(SP);Gnam All
lea MOSTRO1,A0               ;Ind Str Cornacchia
move.w 16(a0),d0             ;Prendo Delta X
add d0,(A0)                  ;Sommo Delta X
```

```
jsr POS                         ;Faccio Agg posiz.
lea MOSTRO3,a0                  ;Str Rete in A0
cmp.b #1,FIREFLAG               ;Lancio in Corso?
bne NOFIRE                      ;No allora NOFIRE
FIRE:                           ;Si
subq #4,2(a0)                   ;YRete = YRete - 4
jsr POS                         ;Faccio agg pos
bra NEXT                        ;Fai il resto
NOFIRE:                         ;Killo RETE
move.w #$140,(a0)               ;Gli do una X di
jsr POS                         ;320 (fuori area)
NEXT:                           ;Faccio il resto
lea MOSTRO1,a0                  ;Str Cornacchia
jsr JOY                         ;Test Joystick
jsr TESTCOLLISION               ;Test Collisioni
jsr MKL                         ;Bingo Bongo
MOVEM.L (SP)+,d0-d7/a0-a6;Scarico
dc.w $4EF9                      ;= JMP
OLDIRQ:                         ;Durante es prg
DC.L 0                          ;metto ind a cui
                                ;Jumpare


POS:
movem.l d0-d7/a0-a6,-(sp);GNAM GMAM
move.l 4(a0),a1                 ;Ind sprite in mem
move.w (a0),d0                  ;X in d0
move.w 2(a0),d1                 ;Y in d0
move.w d0,d2                    ;Copio d0
asR.w #1,d2                     ;Div 2
add.b #64,d2                    ;+ delta X
add.b #44,d1                    ;+ delta y
move.b d1,(a1)                  ;VSTART
addq #5,d1                      ;ricavo VEND
Move.b d1,2(a1)                 ;VEND
move.b d2,1(a1)                 ;Pari o dispari?
and.b #$01,d0                   ;Chi vincera`??
or.W #0,d0                      ;Lascio Word CTRL
move.b d0,3(a1)                 ;Salvo Word CTRL
movem.l (sp)+,d0-d7/a0-a6;Vohaahaah ,sput
rts                             ;Torna da Zio CORNY


JOY:
movem.l d0-d7/a0-a6,-(sp);GRUPHF
lea MOSTRO2,a0                  ;Str Dorarge a0
move.w JOY1DAT,d0               ;Test Jpy
btst #1,d0                      ;Destra ?
bne DESTRA                      ;Si,vai DESTRA
btst #9,d0                      ;Sinistra ?
bne SIN                         ;Si,vai SIN
GREYLORD:                       ;NedestraNesinistra
cmp.b #0,FIREFLAG               ;Lancio in corso?
beQ FIRECHECK                   ;No allora vuoi spararti?
sub.w #1,FIRECONT               ;SI dec FIRE CONTER
cmp.w #0,FIRECONT               ;0 ? NO allora non puoi
bne NOPRESS                     ;sparare.
move.b #0,FIREFLAG              ;Puoi spararti
FIRECHECK:                      ;
btst #7,CIAA                    ;L`hai fatto?
bne NOPRESS                     ;No allora NOPRESS
FIREPRESS:                      ;Hai sparato
move.b #1,FIREFLAG              ;Lancio in corso
move.w #40,FIRECONT             ;40 Seg FIRE CONTER
lea MOSTRO3,a1                  ;Struttura rete in a1
move.l (a0),(a1)                ;Posiziono nella dim
```

```
NOPRESS:                ;spazio_temp di Dorarge
jsr POS                 ;Aggiorna words ctrl
movem.l (a7)+,d0-d7/a0-a6;Bleah
rts                     ;Skippa via

DESTRA:                 ;
add.w #8,(a0)           ;+8
bra GREYLORD            ;


SIN:                    ;
sub.w #8,(a0)           ;-8
bra GREYLORD            ;


TESTCOLLISION:          ;
movem.l d0-d7/a0-a6,-(sp);Altra abboffata
move.w CLXDAT,D0        ;CLXDAT in d0
btst #10,d0             ;Collisione?
beq NOCOLL              ;No non c`e`
lea MOSTRO1,a0          ;SI c`e` allora
add.w #1,16(a0)         ;incremento dx
move.l #0,(A0)          ;della cornacchia
NOCOLL:                 ;e la posiziono a 0,0
movem.l (sp)+,d0-d7/a0-a6;Allegerisciti
rts                 ;vai in via dei matti n.0

MKL:
movem.l d0-d7/a0-a6,-(sp);Burry
move.l PIANO1,a0        ;
moveq #7,d0             ;
ROTT:                   ;
move.l $1e28(a0),d1     ;
ror.l #1,d1             ;
move.l d1,$1e28(a0)     ;
add.l #40,A0            ;
dbra d0,ROTT            ;
movem.l (sp)+,d0-d7/a0-a6;Purry
rts                     ;Fine burry purry

OLDCOPPER:              ;Puntatore alla
dc.l 0                  ;vecchia copperlist

MEMOGG:
DC.L 0

TABELLAMOSTRI:
DC.L MOSTRO1,MOSTRO2,MOSTRO3

FIRECONT:
DC.W 0

PIANO1:
DC.L 0


COPPERLIST:
dc.w $0180,$0000,$0182,$000f ;C0,C1 ( C=COLOR)
dc.w $01a2,$00ff,$01a4,$0fff
dc.w $01a6,$0f00             ;C 17,18,19
dc.w $008e,$2c81,$0090,$2cc1 ;DIWSTART,DIWSTOP
dc.w $0092,$0038,$0094,$00d0 ;DDFSTART,DDFSTOP
dc.w $0108,$0000,$0102,$0000 ;MODULO,SCROLL
BITPLANE1:
dc.w $00e0,$0005,$00e2,$0000 ;BPLOPOINTERS
dc.w $0100,$1200  ;un bitplane (320*255)
SPRITEPTR0:
```

```
dc.w $0120,$0001,$0122,$8000 ;1(0) DMA
SPRITEPTR1:
dc.w $0124,$0001,$0126,$9000 ;2(1) DMA
SPRITEPTR2:
dc.w $0128,$0001,$012a,$9000 ;3(2) DMA
SPRITEPTR3:
dc.w $012c,$0001,$012e,$9000 ;4(3) DMA
SPRITEPTR4:
dc.w $0130,$0001,$0132,$9000 ;5(4) DMA
SPRITEPTR5:
dc.w $0134,$0001,$0136,$9000 ;6(5) DMA
SPRITEPTR6:
dc.w $0138,$0001,$013a,$9000 ;7(6) DMA
SPRITEPTR7:
dc.w $013c,$0001,$013e,$9000 ;8(7) DMA
dc.w $ffff,$fffe       ; End Copper List

SPRITE:
dc.w $9040,$9500
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0f0f,$00ff
dc.w $0000,$0000

MOSTRO1:
;IL MIO NOME E` CRA CRA
DC.W 0              ;X
DC.W 0  ;Y
DC.L 0              ;MEM RELAE
DC.L SPRITE         ;MODELLA
DC.L SPRITEPTR0 ;SEG COPPER PRG.
DC.W 1              ;DELTA X
DC.W 1

MOSTRO2:
;IL MIO NOME E` DORARGE
DC.W 1              ;X
DC.W 150            ;Y
DC.L 0              ;MEM REALE
DC.L SPRITE         ;MODELLO
DC.L SPRITEPTR2 ;SEG COPPER PRG.

MOSTRO3:
;BUM BUM BUM
DC.W 81             ;X
DC.W 10             ;Y
DC.L 0              ;MEM REALE
DC.L SPRITE         ;MODELLO
DC.L SPRITEPTR4 ;SEG COPPER PRG.

even

GFXNAME:
dc.b 'graphics.library',0

FIREFLAG:
DC.B 0
```

VIDEO - EFFETTI

Impariamo ad usare alcune librerie di Amiga ed approfittiamone per saperne di più su questo potente computer; anche da Basic

di Davide Martinenghi

Parliamo di Basic, ma non illudetevi che siano argomenti semplici

Il programma è solo una dimostrazione di come anche il Basic possa sfruttare le più importanti peculiarità di Amiga. Si limita a dichiarare qualche **funzione**, aprire una **libreria** e usare le sue caratteristiche in modo insolito.

Dopo aver digitato (e salvato su disco) il programma, è preferibile che chiudiate la finestra *List* e rendiate visibile la *Title Bar* dello schermo. Dopo il *Run* inizierà una rassegna di effetti di Intuition.

Dopo qualche istante, necessario per l'accesso alla libreria, apparirà una finestra che, come se fossa mossa da un mouse, si sposterà di alcune posizioni sul video. Sarà poi creato un nuovo schermo, dietro quello solito di WorkBench, che, mediante due funzioni, verrà posto in primo piano e quindi nuovamente dietro, come se aveste selezionato i Back gadget dei due schermi.

Verrà quindi effettuato uno scrolling verso il basso del WBScreen; in seguito tre rapidi flash illumineranno il video (molto simili a Beep, ma privi di sonoro).

La finestra spostata in precedenza verrà ora ridimensionata, proprio come se aveste usato il suo Sizing gadget. Apparirà poi uno schermo nero contenente un Alert (avete presente le Guru Meditation?) che vi consente di scegliere, con i mouse buttons, se scrivere **ciao** oppure **salve**.

Effettuata questa operazione, dopo aver premuto un tasto, ritornerete nel WBScreen e vedrete apparire due finestre che, se selezionate, schermo, daranno luogo a due scritte diverse; cliccando in quella di sinistra farete apparire una nuova figura del puntatore del mouse (la famosa freccetta).

Il programma richiede che sia presente, nella directory corrente, il file **Intuition.bmap** che potrete ottenere con il programma sul disco **Extras** nel cassetto **BasicDemos**. Per fare ciò dovrete:

- abilitare il vostro disco alla scrittura (coprendo la tacchetta) e metterlo da parte.
- inserire il dischetto Extras1.3 ed aprire la finestra relativa al cassetto **BasicDemos**
- cliccare sul programma **ConvertFD**
- alla domanda **"Enter name..."** scrivere **Extras1.3:FD1.3/intuition_lib.fd** (se, ovviamente, il dischetto Extras ha il nome Extras1.3, altrimenti provvedete alle dovute modifiche).
- alla successiva domanda "**Enter name...**" scrivere **nomedisco:intuition.bmap**

Dopo (a volte numerosissimi) scambi di dischetti, il programma ConvertFD avrà creato il file **intuition.bmap** sul vostro dischetto e tutte le potenzialità di questa libreria saranno a vostra disposizione. Anche se però, a questo punto, vi trovate in ambiente Basic, vi conviene uscire, "chiudere" il Basic e riaprirlo dal vostro dischetto per evitare errori del tipo "File not found".

E' indispensabile, ora, conoscere la differenza tra **funzione** e **comando**: le funzioni vanno sempre dichiarate prima di aprire la libreria e hanno un formato diverso dai comandi. In genere:

variabile = funzione (parametri)

CALL comando (parametri)

"variabile" contiene un valore che solitamente annuncia l'esito della chiamata: 1 = OK; 0 = errore. Ma analizziamo il programma.

Il primo comando è **MoveWindow**, il cui effetto è quello di spostare una particolare finestra di un certo delta. I parametri sono:

CALL MoveWindow& (Window, DeltaX, DeltaY)

dove **Window** è il puntatore a una "struttura" window (il problema è stato superato utilizzan-

do la funzione WINDOW(7) che restituisce tale puntatore relativo alla finestra corrente);

DeltaX e **DeltaY** sono i parametri, che possono essere anche negativi, che indicano di quanto spostare la finestra, a patto di non uscire dai limiti dello schermo.

E' poi la volta di due funzioni, **WBenchToFront** e **WBenchToBack**, che non richiedono parametri e che spostano rispettivamente in primo piano e dietro il WBScreen:

a& = WBenchToBack&
b& = WBenchToFront&

dove **a&** e **b&** contengono **1** se l'operazione ha successo, altrimenti **0**.

Nel corso del programma si preleverà anche **WB**& che è il puntatore alla struttura del WBScreen, utilizzando la funzione **OpenWorkBench** che non richiede parametri. Saremo ora in grado di modificare anche il WBScreen, avendone il puntatore. Innanzitutto verrà chiamato il comando MoveScreen, in tutto simile al comando MoveWindow, con l'unica differenza che su questo non ha effetto il DeltaX:

CALL MoveScreen& (Screen, DeltaX, DeltaY)

dove Screen è il puntatore ad una struttura screen (come WB&).

Potremo anche effettuare dei "flash" sul WBScreen con il comando DisplayBeep:

CALL DisplayBeep& (Screen)

dove Screen è il solito puntatore, ma se = 0 indica che tutti gli schermi devono avere un flash.

Conoscendo ormai le informazioni basilari, descriveremo il resto con rapidità.

CALL SizeWindow& (Window, DeltaX, DeltaY)

che aggiunge il delta specificato alla dimensione della finestra;

CALL SetWindowTitles& (Window, WindowTitle, ScreenTitle)

dove WindowTitle e ScreenTitle sono i puntatori ai nomi della finestra e dello schermo (quando viene selezionata), motivo per cui si deve usare SADD(nomestringa$), dove la stringa termina con CHR$(0).

Per quanto riguarda gli Alert (messaggi in stile Guru Meditation):

a& = DisplayAlert& (AlertNumber, String, Height)

dove **AlertNumber** è un numero che indica il tipo di Alert (0 = recuperabile), **Height** indica l'altezza del rettangolo rosso lampeggiante, **String**, più complesso, è composto da due bytes che indicano la posizione X in cui scrivere il messaggio, 1 byte per la posizione Y, il messaggio effettivo e un CHR$(0), più un eventuale byte che, se diverso da 0, indica che c'è un'altra stringa. Chiaramente in Basic si deve "passare" il puntatore, usando SADD. Si noti che se è stato premuto il tasto sinistro a& contiene 1, altrimenti 0.

Si fa uso di altri due comandi: **SetPointer** e **ClearPointer** che, rispettivamente, modificano la freccetta del mouse e la ripristinano.

CALL ClearPointer&

...che non richiede parametri;

CALL SetPointer& (Window, Pointer, Height, Width, XOffset, YOffset)

...dove Height indica l'altezza della freccetta, Widt è la larghezza, XOffset e YOffset indicano le coordinate del "punto significativo" della freccetta e Pointer è il puntatore ad un insieme di coppie di word (gruppo di 16 bit) che contengono la definizione della forma della freccetta. Si stabilisce il colore di ciascun punto accoppiando verticalmente i bit delle due word in una coppia. Ad esempio, i valori decimali...

65278 e 33026

...nel programma indicano:

1111111011111110
1000000100000010

il che significa che ci saranno 16 punti orizzontali dei colori numerati, rispettivamente, con 3 (prima coppia verticale: 11), 2 (seconda coppia verticale: 10), 2 (idem), 2, 2, 2, 2, 1 (01), 2, 2, 2, 2, 2, 2, 3, 0 (00) cioè sfondo.

In Basic l'insieme di word è ottenuto con un VARPTR(a%(0)), ma poichè un numero intero (%) non può superare 32767, cioè 127 * 256 + 255 (in quanto da 128 * 256 + 0 iniziano i numeri negativi) si è usata la formula a% = a65536&.

Il programma usa inoltre convenzionalmente il segno **&** dopo ogni nome di comando e di funzione.

VIETATO AI MINORI

E' probabile che i neo-utenti di computer non riescano a ben comprendere l'utilità della procedura software descritta nelle presenti pagine. Questa è infatti destinata a coloro che sono già padroni delle principali tecniche di programmazione e desiderano approfondire il modo in cui un elaboratore organizza i vari dati all'interno della memoria.

Il programma, pertanto, rappresenta un invito ai lettori più esperti, soprattutto a coloro che, considerando il listato come una base di partenza, riescano a pervenire a procedure più interessanti e capaci, magari, di offrire applicazioni di più ampio respiro.

Se hai incominciato da poco, comunque, non scoraggiarti! Tutti coloro che, oggi, vantano una particolare competenza nel campo dell'informatica hanno iniziato con un banale Print "Pippo".

Perchè non dovresti riuscire anche tu?...

```
'intuition effects
'di Davide Martinenghi
DIM a%(29)
FOR k = 0 TO 29: READ a: ' Legge i dati del puntatore
IF a > 32767 THEN a%(k) = a - 65536&: ELSE a%(k) = a: 'e li mette in a%()
'se avete l'espansione digitate la seguente riga
'POKEW 400000& + k*2, a%(k): '(gli sprites solo nella CHIP MEMORY)
NEXT
DECLARE FUNCTION DisplayAlert& LIBRARY
DECLARE FUNCTION OpenWorkBench& LIBRARY
DECLARE FUNCTION WBenchToFront& LIBRARY
DECLARE FUNCTION WBenchToBack& LIBRARY
LIBRARY "intuition.library"
MENU 1, 0, 0, "": MENU 2, 0, 0, "": MENU 3, 0, 0, "": MENU 4, 0, 0, "": 'Elimina i menu
WINDOW 1, "Finestra", (100, 50) - (400, 150)
PRINT "Muoviti!"
CALL MoveWindow& (WINDOW(7), 100, 20): 'Muove la finestra di 100, 20
FOR k = 1 TO 1000: NEXT: 'Attesa
CLS: PRINT "WorkBench to Front"
SCREEN 1, 640, 200, 2, 2: ' Crea un nuovo schermo
WINDOW 2, "Finestra", (200, 70) - (500, 170), , 1: 'e ci mette una finestra
PRINT "WorkBench to Back"
a& = WBenchToBack&: FOR j = 1 TO 2000: NEXT: ' Pone il WBScreen dietro
b& = WBenchToFront&: FOR j = 1 TO 2000: NEXT: 'Pone il WBScreen davanti
WB& = OpenWorkBench&: 'Preleva il puntatore al WB screen
WINDOW OUTPUT 1: CLS: PRINT "Scorri!"
FOR k = 1 TO 250
CALL MoveScreen& (WB&, 0, 1): ' Muove il WBScreen in giu'
NEXT
CALL MoveScreen& (WB&, 0, - 250): ' e lo riporta su
SCREEN CLOSE 1: 'Chiude lo schermo aperto prima
CLS: PRINT "Beep beep"
FOR k = 1 TO 3: ' Per 3 volte
CALL DisplayBeep& (WB&): FOR j = 1 TO 2000: NEXT: 'fa un flash sul video
NEXT
WINDOW OUTPUT 1: 'output alla finestra 1
CLS: PRINT "Cresci!"
CALL SizeWindow& (WINDOW(7), 90, 20): 'Aggiunge alla finestra un delta di (90, 20)
FOR k = 1 TO 5000: NEXT: 'Attesa
SCREEN 1, 640, 200, 2, 2: ' Riapre un nuovo schermo
WINDOW 2, "", (0, 16) - (600, 180), 0, 1: ' e ci rimette una finestra
PALETTE 0, 0, 0, 0: PALETTE 1, 0, 0, 0: ' I colori sono
PALETTE 2, 0, 0, 0: PALETTE 3, 1, 1, 1: COLOR 3: ' nero e bianco
m$ = CHR$(0) + CHR$(170) + CHR$(20) + "Left Right" + CHR$(0) + CHR$(1)
m$ = m$ + CHR$(0) + CHR$(170) + CHR$(36) + "Ciao Salve" + CHR$(0)
'Crea una stringa di due righe con coordinate specifiche (170, 20) e (170, 36)
h& = DisplayAlert&(0, SADD(m$), 60): 'Crea un Alert con la stringa m$
PRINT: PRINT: PRINT SPC(35);
IF h& = 0 THEN PRINT "Salve";: ELSE PRINT "Ciao";
'Stampa il messaggio corrispondente a seconda del tasto mouse premuto
LOCATE 10, 10: PRINT "Premi un tasto"
1 x$ = INKEY$: IF x$ = "" THEN 1: 'Attende la pressione di un tasto
SCREEN CLOSE 1: ' e richiude lo schermo
WINDOW 1, , (0, 10) - (300, 186): ' Crea una finestra senza nome
WINDOW 2, , (318, 10) - (600, 186), , - 1: 'Crea una finestra senza nome
a$ = "MyWindow" + CHR$(0): b$ = "MyScreen" + CHR$(0)
c$ = "MiaFinestra" + CHR$(0): d$ = "MioSchermo" + CHR$(0): 'Crea 2 coppie di stringhe
WINDOW OUTPUT 1: 'Da' l'output alla prima finestra
CALL SetWindowTitles& (WINDOW(7), SADD(a$), SADD(b$))
'Da' il nome alla finestra (a$) e allo schermo (b$)
CALL SetPointer& (WINDOW(7), VARPTR(a%(0)), 13, 16, - 8, - 6)
'se avete l'espansione cancellate la riga sopra
'e digitate quella sotto
'CALL SetPointer& (WINDOW(7), 400000&, 13, 16, - 8, - 6)
'Assegna la nuova forma del puntatore della window 1
WINDOW OUTPUT 2: 'da' l'output alla seconda finestra
CALL SetWindowTitles& (WINDOW(7), SADD(c$), SADD(d$))
'Da' il nome alla finestra (c$) e allo schermo (d$)
PRINT "Seleziona una delle due."
PRINT "Oppure clicca qui e": PRINT "premi spazio per uscire"
2 x$ = INKEY$: IF x$ < > " " THEN 2
WINDOW CLOSE 2: ' Chiude la window 2
WINDOW 1, "IntuiFX", (0, 0) - (617, 186): 'e ridisegna la window 1
CALL ClearPointer (WINDOW(7)): 'Ripristina il puntatore
LIBRARY CLOSE: 'Chiude la libreria
MENU RESET: 'Ripristina i menu
END
DATA 0, 0, 4064, 4064, 12568, 12312
DATA 24844, 24588, 49414, 49158, 33026
DATA 32770, 33026, 32770, 65278, 33026
DATA 33026, 32770, 33026, 32770, 49414
DATA 49158, 24844, 24588, 12568, 12312
DATA 4064, 4064, 0, 0
```

FORMAT

NOME. Nome che si intende assegnare al dischetto. E' ammesso un numero massimo di 30 caratteri, da racchiudere tra doppi apici nel caso sia presente qualche spazio al suo interno. Proprio per questa fastidiosa "incombenza" (presente anche in altri comandi), è prassi comune fondere in un unico insieme eventuali nomi composti, o adoperare il carattere di sottolineatura (_) al posto dello spazio (per esempio MioDisco, oppure Mio_Disco).

NOICONS. Opzionale.
Se inserito nella riga di comando, il disco formattato non conterrà la directory (e relativa icona- cestino) Trashcan, non escludibile se invece si adopera l'opzione Initialize dal menu del Workbench.

Format. Impartito da solo, visualizza una schematica descrizione della sua sintassi completa.
Drive, Name. Parole chiave obbligatorie, da inserire sempre nella sintassi del comando.

FORMAT DRIVE unità NAME nome NOICONS QUICK

Unità Descrizione della periferica nella quale si trova il floppy da formattare. In pratica, può corrispondere a Df0: (drive interno negli A500), Df1:, Df2:, eccetera.

QUICK. Opzionale. Come una traduzione letterale del termine fa già intuire (quick = veloce), consente l'inizializzazione rapida di un disco, a patto che questo non sia "vergine". Questo parametro si rivela molto utile quando si intende cancellare il contenuto preesistente di un dischetto senza ricorrere alla (relativamente) lenta procedura di formattazione. In pratica, vengono formattati solo la traccia contenente le indicazioni necessarie al riconoscimento della directory principale (Root) ed il cosiddetto Boot Block (quello che fa avviare automaticamente un dischetto se lo si inserisce nel drive a sistema "inattivo"). Dopo il trattamento, il floppy risulterà vuoto a tutti gli effetti, esattamente come se si fosse ricorsi alla formattazione globale.

ESEMPI

Incominciamo dal caso più semplice ed usato dalla maggior parte degli utenti.

Format Drive Df1: Name MIO Noicons

Inizializza il disco presente nel secondo drive, assegnandogli il nome "MIO", senza creare la directory Trashcan. Il nome del disco potrà essere adoperato in altri comandi nella sequenza maiuscolo/minuscolo che più si preferisce, mentre nelle visualizzazioni di sistema (tanto in Workbench che in Shell) verrà sempre riprodotto così come lo si è impostato nel parametro Nome. Nel nostro esempio, volendo modificare (dopo la formattazione) MIO nei corrispondenti caratteri minuscoli, si dovrà ricorrere al comando Relabel da Shell, oppure all'opzione Rename da ambiente Workbench.

Format Drive Df0: Name Mio Quick

Prepara all'uso il floppy inserito nell'unità a dischi principale, tentando la formattazione veloce. Il dischetto conterrà la directory Trashcan.

Nota bene

Le parole chiave di ogni comando sono rappresentate in maiuscolo e vanno (eventualmente) adoperate così come sono.
In minuscolo, sono invece riprodotti i parametri che vanno ridefiniti dall'utente.

AMBIENTI CONTRAPPOSTI

Il programma **Format**, a differenza della maggior parte degli altri **comandi** del Dos, è memorizzato nella directory **System** del disco **Workbench**. Il fatto che sia possibile richiamarlo direttamente è dovuto alla presenza, nella **startup-sequence** del dischetto (presente nella directory **S**), di una istruzione **Path System**.
La directory system, d'altra parte, è spesso uno di quegli elementi che si è portati a sopprimere nella creazione di un proprio disco di lavoro, magari trasferendo Format nella directory C.
In tal modo, si potrà sempre disporre facilmente del comando digitandolo da **Shell** senza alcuna indicazione di percorso, ma occorre prestare attenzione alle conseguenze rilevabili da Workbench: la sua opzione **Initialize**, infatti, adopera anch'essa Format, andandolo però a cercare sempre e solo nella directory System. Se non lo trova, genera un messaggio di errore.
Di contro, pur mantenendo il comando nel suo luogo... d'ordinanza, se si è modificata la Startup-sequence eliminandone il comando Path, o si è impartito direttamente da Shell per qualche motivo il comando **Path Reset** (vedi descrizione di Path sul numero scorso), si otterrà una segnalazione di errore **Unknown Command** invocando Format da ambiente Dos.

ERRORI

Se si esclude l'eventualità (descritta a parte) che il comando non venga rintracciato dal sistema, la più frequente possibilità di errore è legata ad una scorretta digitazione della lunga e prolissa sintassi di questo comando: inspiegabile, in tal senso, il motivo della inamovibilità delle parole chiave **Drive** e **Name**.
Questo tipo di errore viene segnalato dalla semplice visualizzazione della corretta sintassi, esattamente come se si fosse digitato **Format** senza altri parametri.

Format failed - bad sector header...

Questo messaggio indica che si è tentato di formattare con l'opzione Quick un disco non Dos, ovvero mai formattato prima da AmigaDos. Per ovviare, è sufficiente ripetere l'operazione senza il parametro Quick.

In questa eventualità, basterà comunque digitare il suo percorso completo, ovvero...

Sys:System/Format

...o anche solo...

System/Format

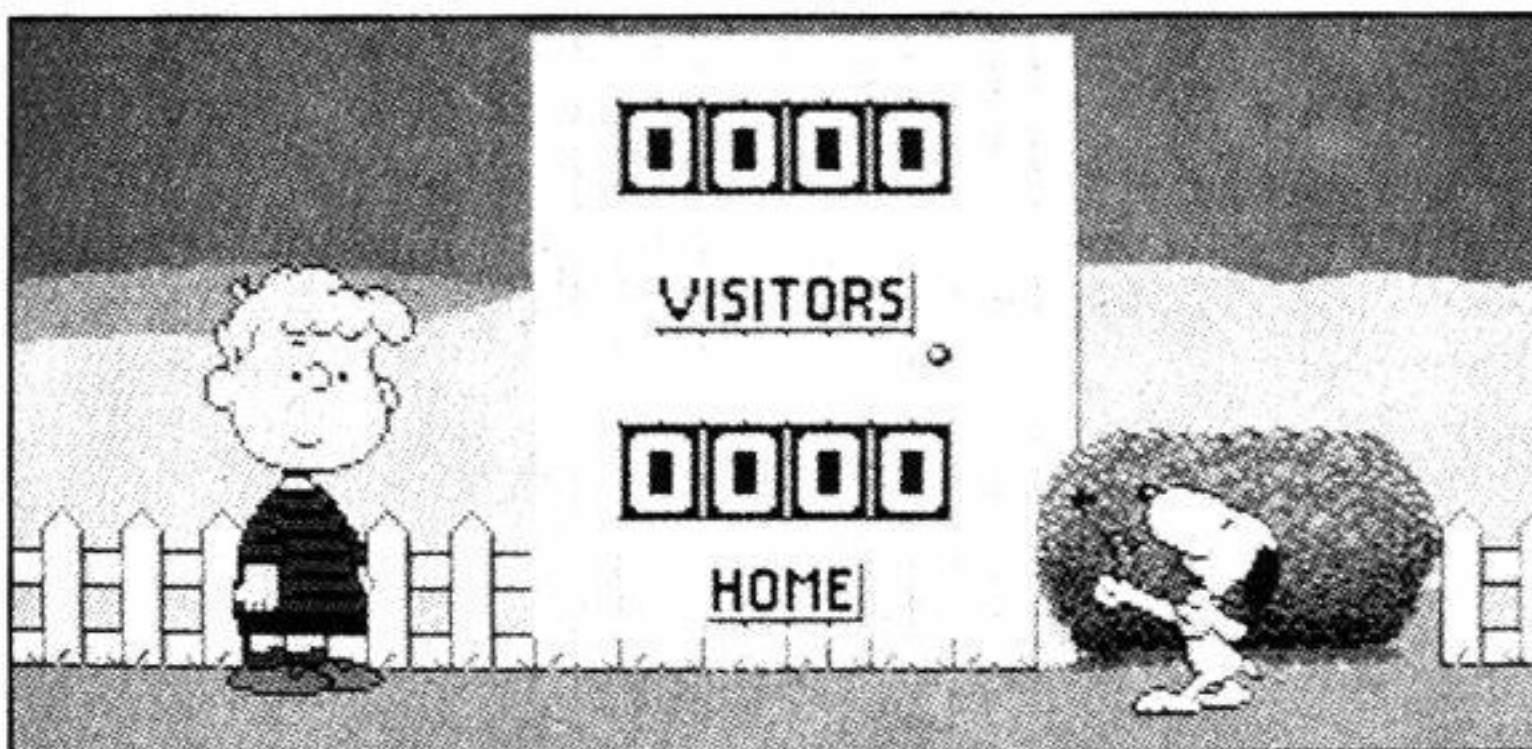
...se ci si trova già nella directory principale del disco adoperato per il boot.

ATTENZIONE A...

Non dimenticare il sibolo doppio punto (:) nella descrizione della periferica (**df1:** e non solo **df1**).

L'inizializzazione di un dischetto comporta la perdita di tutti i suoi dati. Per tale motivo, prima che l'operazione abbia inizio, il sistema richiede sempre conferma dell'operazione tramite appositi Requester.

Per arrestare il procedimento si può anche ricorrere alla pressione di **Ctrl+C**, ma, una volta avviata la formattazione, non c'è più nulla da fare: il contenuto del dischetto risulterà in ogni caso irrecuperabile.

Prudenza ed attenzione eviteranno spiacevoli equivoci.

FORMAT. Prepara un supporto magnetico (floppy o hard disk) in modo che AmigaDos possa "riconoscerlo" e sfruttarlo. Tale processo è comunemente definito Inizializzazione, o anche Formattazione, e risulta indispensabile prima di qualunque altra operazione che coinvolga l'accesso ad un disco. L'unica eccezione a tale regola è legata all'uso del comando Diskcopy, che effettua automaticamente la formattazione mentre svolge il suo compito. Nel formato standard di AmigaDos, un floppy disk viene suddiviso in 80 "cilindri", numerati da 0 a 79, ognuno dei quali composto da 22 "blocchi" di 512 byte ciascuno, per una capienza totale di 880 Kbyte. In questa sede, non viene esaminata l'azione di Format in rapporto all'uso di dischi rigidi.

PROTECT

S

Flag **Script**. Se aggiunto dopo la specifica del nome, indica che ci si trova in presenza di un **Batch File**, o File comandi che dir si voglia, composto quindi da una serie di comandi Dos. Questo tipo di file, come noto, non è altro che un testo in Ascii, normalmente eseguibile da Shell (o Cli) solo tramite il comando **Execute**.
Settando però questo attributo nel nome del file, basterà invocare direttamente il suo nome da Shell per mandarlo in esecuzione. Si provi a verificarlo procedendo come segue: con **ED** si editi un file di nome **Test** (comando: ED test) contenente la sola istruzione...

Echo "File eseguito!"

...(**Escape + X** per uscire da Ed memorizzando il file). Quindi, si impartisca **Protect Test S** e si controlli con **List** che al file di nome test sia associato il flag Script. Se tutto è in regola, si provi ora a digitare **Test** (+ Return): la stringa **File eseguito** dovrebbe apparire esattamente come se si fosse lanciato un normale file-programma!
Naturalmente, il comando **Execute** deve in ogni caso trovarsi nella directory **C** del disco di sistema, in quanto sarà sempre "lui" ad operare, anche se non siamo stati costretti ad invocarlo.
Nel caso appena visto, poichè si è usato il solo flag **S**, il file non potrà essere cancellato o riscritto, a meno che non si reimpostino anche i flag **W** e **D**, o si sostituisca il comando prima digitato con (per esempio) **Protect SWD**.

P

Bit **Pure**. Indica che il programma è "rientrante", come spiegato dal manuale Dos in dotazione ad Amiga alla voce **Resident**. A meno di non essere già sufficientemente esperti, non è consigliabile settare da sè questo flag (si veda comunque il manuale per la procedura adatta a stabilire se un file è rientrante). Opportuno, invece, accertarsi sempre con **List** che il bit **P** sia associato ad un file che si intende rendere residente.

R

Read. Se non è impostato, impedisce l'accesso al file in questione. Il che, si badi, non vuol dire che (per esempio) un comando Type non funzionerà su di esso.
In effetti, anche eliminandolo, un editing del file (con ED) è ugualmente possibile, ed anzi, ad una successiva memorizzazione dello stesso, il flag **E** viene reimpostato automaticamente . In pratica, risultano molto più utili gli altri flag di protezione.

PROTECT nome S P R W E D

Nome. Digitare il nome del file del quale modificare gli attributi, eventualmente completato dal suo percorso completo, se non è direttamente raggiungibile dalla directory corrente. Se al comando ed al parametro **Nome** non viene fatta seguire alcuna specifica, tutti le funzioni legate ai bit di protezione verranno inibite.

W

Write. Se un comando Protect non comprende l'attributo W (e quindi lo disabilita), viene inibita la possibilità di accedere in scrittura allo stesso file, ad esempio dopo averlo modificato con un editor, o per aggiornarne il contenuto con un Database.

E

Executable. L'efficacia di questo bit è limitata ai file programma, che non vengono più riconosciuti come eseguibili dal dos se manca l'attributo E. In tale eventualità, tentando di lanciare il programma, si otterrà solo una segnalazione **Execute permission not set** seguita dal più tradizionale messaggio: **File is not an object module**.

D

Deletable. Non inserendo la specifica **D** dopo un comando **Protect**, si renderà il file interessato non cancellabile (fino, ovviamente, ad un nuovo Protect che riabiliti tale possibilità).

PROTECT nome + - flag ADD SUB

Con la prima sintassi, quali che siano i flag inseriti tramite il comando, occorre badare al fatto che vengono in realtà modificati **tutti** gli attributi: una istruzione...

Protect MioPrg RW

...per esempio, preciserà che il file sarà accessibile in lettura e scrittura, ma nel contempo azzererà gli altri flag.
Volendo agire solo su uno in particolare, si renderebbe quindi necessario appurare prima quali sono i flag attivi con List, per poi operare di conseguenza.

In alternativa, si può ricorrere alla **seconda forma sintattica**, molto più comoda, che consente di "aggiungere" o eliminare singoli flag senza modificare i rimanenti.
Per far ciò, basta adoperare il simbolo più (+) (per inserirlo) oppure meno (-) (per inibirne la funzione) prima della specifica del flag (S, P, R, W, E, D come nella prima forma sintattica) e senza alcuno spazio tra il segno e la lettera.
O anche (meno comodo) posporre **Add** (aggiungi) oppure **Sub** (elimina) alla lettera che precisa il flag.

PROTECT

Consente di modificare, impostare o rimuovere i cosiddetti **bit di protezione** associati ad un file (oppure ad una directory), in pratica corrispondenti ad una serie di **attributi** che abilitano o inibiscono particolari operazioni consentite sui files da considerare.
Tali attributi sono rilevabili impartendo un **List** del file in questione, ed osservando quanto visualizzato tra la sua **dimensione** in byte e la **data** ad esso associato.
In particolare, si potrà constatare la presenza di una serie di caratteri (nella maggior parte dei casi "**----rwed**"), la cui presenza indica che la funzione ad essi associata è attiva. In caso contrario, alla singola lettera viene sostituito un trattino.

ESEMPI

Protect Ram:test

Elimina tutti gli attributi legati al file **Test**, memorizzato nella **Ram Disk**.
Impartendo...

List Ram:test

...al posto dei flag saranno presenti otto trattini ("- - - - - - - -").

Protect Df1:miobatch SRW

Il file **Miobatch**, presente nel disco inserito nel secondo drive, potrà essere mandato in esecuzione direttamente anche se si tratta di un file comandi (**S**). Potrà inoltre essere letto e riscritto, ma non cancellato (**D** non è specificato, e quindi verrà eliminato).

Protect Prog -D+S

Il file **Prog** non potrà essere cancellato, e viene contemporaneamente convalidato come Script.
Gli altri flag rimangono inalterati.

ERRORE PIU' FREQUENTE

Protect failed.

Al 99,99 per cento, si è adoperato un nome di file scorretto, con ogni probabilità per un errore di digitazione, o senza precisarne il percorso: in pratica, il Dos avverte che non ha trovato il file.

PROTECT

La sintassi del comando prevede anche l'uso facoltativo delle parole chiave **File** e **Flags** poste rispettivamente prima e dopo il parametro **Nome**. Da dimenticarne in fretta l'esistenza. I parametri **SPRWED** possono anche essere associati in una stessa linea di comando, e la loro presenza indica che quel particolare flag deve essere aggiunto, mentre quelli non specificati verranno disabilitati.

DELETE

DELETE
Impartito da solo, produce una semplice segnalazione "No file to delete" (nessun file da cancellare).

nome Il nome del file o dei files da rimuovere, eventualmente preceduti dal loro percorso completo.
Si possono specificare fino a **10** files in una stessa riga di comando, separati tra di loro da uno spazio. E' anche possibile inserire nel nome un **Pattern**, ovvero uno dei simboli che consentono di riferirsi a più files (**#?**, **?**, eccetera). **Nome** può indicare una (o più) directory, nel qual caso (in assenza di altri parametri) verrà cancellata solo se già vuota.

ALL. Aggiungendo questa **keyword** (parola chiave) quando si applica Delete al nome di una directory, questa viene cancellata assieme a tutto il suo contenuto, ivi comprese eventuali subdirectory in essa annidate (**adoperare con cautela!**).
Se aggiunta dopo il nome di un file, viene semplicemente ignorata, ed il file regolarmente rimosso.

DELETE nome ALL QUIET

QUIET. Opzionale. Elimina la visualizzazione su schermo del procedere delle operazioni, con un guadagno in termini di velocità tutto sommato non realmente significativo. Provare, e decidere se la supplementare fatica digitatoria ha un reale ritorno pratico.

DELETE

Rimuove da un disco fisico o virtuale (per esempio **Ram Disk**) files oppure ditrectory in esso contenuti.

Funzione analoga viene svolta, in ambiente Workbench, dalla opzione **Discard** del menu principale, che peraltro risulta completamente "sganciata" dal comando Dos, tanto da poter svolgere la sua azione anche in assenza del file Delete, memorizzato nella solita directory C.

Delete, come del resto un po' tutto l'ambiente Shell, risulta però alquanto più versatile del "benchiano" Discard, legato indissolubilmente alla presenza delle icone associate ai files.

Il comando Dos, inoltre, consente di rimuovere anche la directory (e relativa icona) Trashcan, inamovibile da Workbench.

ESEMPI

Non tutti sanno che è possibile cancellare, con un solo comando, più files presenti in device diversi. Il seguente comando, infatti...

Delete Df1:c/status Ram:miofile

...cancella il file di nome **Status** dalla directory **C** della seconda unità a dischi, nonchè **Miofile** presente nella **Ram Disk**.

Delete Sys:Utilities All

Elimina tutti i files ed eventuali subdirectory inclusi nella directory **Utilities** del disco di sistema, dopodichè rimuove anche la directory stessa.

Delete Utilities/#?

L'esito è simile a quello dell'esempio precedente: cancella tutti i files della directory **Utilities** (che deve in questo caso essere accessibile dalla directory corrente, non essendo specificato alcun percorso), ma la directory non viene rimossa.

Si badi che in questo caso, se all'interno della directory Utilities fossero presenti delle subdirectory, queste ultime rimarranno inalterate, così come gli eventuali files in esse memorizzati.

ATTENZIONE A...

Quando si inseriscono più nomi di files in uno stesso comando **Delete**, occore prestare una certa attenzione nella eventualità che si verifichi un errore. Supponiamo, per esempio, di avere impartito un comando...

Delete File1 File2 File3

...digitando male il nome del **File2**. In questo caso, **File1** verrà cancellato, dopodichè la procedura si arresterà segnalando l'impossibilità di rintracciare File2. **File3**, quindi, non verrà cancellato, così come tutti gli eventuali altri aggiunti in successione.

Il programma Delete, una volta lanciato, esegue il suo compito senza richiedere alcuna conferma: maneggiare con cura!

ERRORI PIU' FREQUENTI

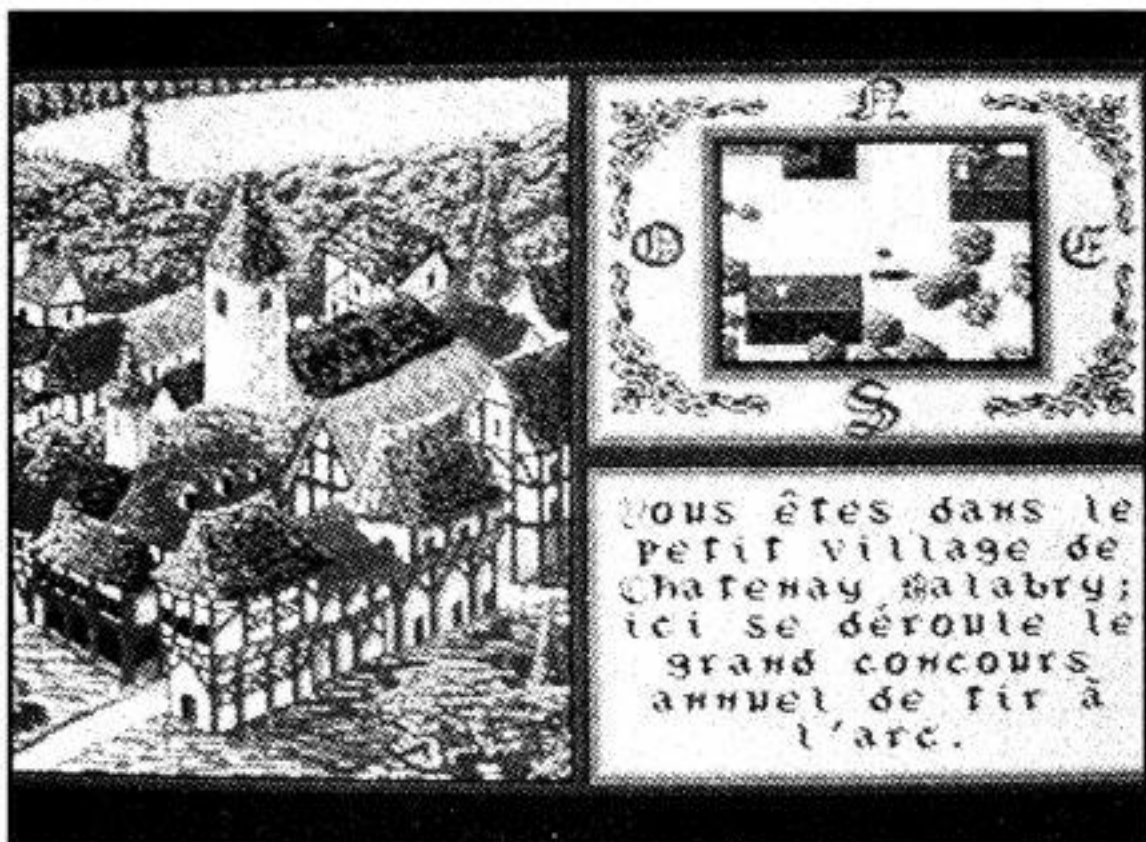
Could Not Get Information...
Si è digitato male il nome del file da cancellare, oppure questo è memorizzato in una directory diversa da quella attuale, e non se ne è specificato il percorso.

Not Deleted - Directory Not Empty
Si è tentato di rimuovere una directory contenente dei files senza aggiungere **All** dopo la specifica del nome.

I CARATTERI SPECIALI

Con alcuni comandi di AmigaDos, è possibile adoperare i cosiddetti **profili di corrispondenza**, ovvero specificare il nome di uno o più files (in quest'ultimo caso purchè posseggano delle comuni ricorrenze di caratteri) senza doverli digitare per intero.
Tale tecnica è definita **Pattern Matching**, ed è legata all'uso di particolari caratteri. Chi proviene da precedenti esperienze "computerecce" maturate sugli otto bit Commodore, oppure ha avuto a che fare con l'ambiente **Ms-Dos**, certamente ha prima o poi sfruttato il classico **asterisco** (*), che indica una qualunque sequenza di caratteri.

Nel mondo Amiga, questo simbolo ha un significato completamente diverso (indica la **console** al momento attiva), ma in compenso esiste un corredo di **wildcards** (i caratteri speciali vengono di solito definiti così) estremamente versatile, anche se non sempre molto semplice da capirsi.
Il ricorso ai profili di corrispondenza non è consentito in tutti i comandi a corredo del disco di sistema: viene infatti limitato ai comandi **Copy**, **Dir**, **Delete** e **List**.

?

Punto Interrogativo.

Può essere posto all'interno del nome di un file ad indicare la presenza, in quel punto, di un **qualsiasi** singolo carattere.
Se esistono più files il cui nome si differenzia solo per quel carattere, verranno tutti coinvolti nell'azione del comando che deve essere eseguito.
Qualche esempio: se si impartisce il comando...
Delete M?lta
...verrà cancellato (dalla directory corrente) un eventuale file di nome **Malta**, ma anche, se presente, uno a nome **Multa**. Si badi, comunque, che il punto interrogativo non sostituisce "l'assenza" di un carattere; tornando all'esempio precedente, un programma **Mlta** non verrà coinvolto nella rimozione.
Nell'ambito dello stesso nome, può anche essere usato più di un carattere speciale.
Ad esempio:
Delete M?lt?
...cancellerà non solo i files prima visti, ma anche **Molto, Molta, Multe, Molti, Miltt**, e così via. Un...
Delete #?
...infine, rimuoverà **tutti** i files della directory corrente (occhio!)

( )

PARENTESI TONDE.

Più che un pattern vero e proprio, questo simbolo può essere considerato come un supporto agli altri, in quanto serve a trasformare in unico elemento di selezione un gruppo di caratteri.
Consideriamo, come esempio, il simbolo di **Pound**.
Questo, come precisato, fa riferimento al singolo carattere che lo segue (vedi descrizione).
Facendolo però seguire da più caratteri racchiusi in parentesi, la selezione riguarderà le eventuali ripetizioni di tutto il gruppo di caratteri, non solo quello subito a destra del cancelletto.
Quindi, per esempio, con...

List #(tam)pone

...verranno listati nomi come **pone** (nessuna ricorrenza), **tamtampone** (2 ripetizioni), eccetera.

Senza le parentesi, il pattern avrebbe vagliato la sola ricorrenza della lettera **T** iniziale.

?

Pound + Punto Interrogativo.

Corrisponde, in pratica, all'**asterisco** degli altri sistemi, ed è certamente la **wildcard** che viene adoperata con maggiore frequenza.
Serve per indicare, nel nome di un file, una qualunque ricorrenza di caratteri, senza alcun vincolo qualitativo e quantitativo. Il che significa, per esempio, che un comando...
List Art#?
...mostrerà a video l'elenco di tutti i files (o Directory, si intende) che cominciano con **Art**, quale che sia il tipo ed il numero dei caratteri che seguono.
Quindi, per esempio, risponderanno al profilo di corrispondenza nomi come **Artista, Articolo, ArtManager, Art.info,** eccetera.
I due simboli possono essere inseriti in qualunque punto del nome, per cui risuterà ad esempio facile eliminare tutte le icone di una directory: basterà impartire...
Delete #?.info
Come per gli altri pattern, anche questa forma può essere inserita più di una volta in un comando, ottenendo una selezione di files il cui nome contenga una stessa ricorrenza al suo interno.
Con...
Delete #?AP#?

...per esempio, si farà agire il comando di delezione su eventuali files di nome **catAPulta, catAPecchia, rAPa, cAPitano**, e così via.

# #

Pound (cancelletto).

Usato da solo, rappresenta un qualunque numero di ricorrenze del carattere che lo segue, come anche una sua eventuale assenza. Più chiaramente: un comando...
Delete #Trotter
...agirà su un file **Trotter** (una ripetizione della lettera **T**), ma anche su **Ttttrotter** (4 ripetizioni) e su **rotter** (0 ricorrenze).

|

Barra Verticale.

Si ottiene, da tastiera, premendo **shift** + **back slash** (\). Ulteriore sofisticazione della scelta tra due (o più) ricorrenze, segnala in pratica una precisa alternativa tra due elementi di selezione.
Più chiaramente, un pattern **X|Y** consente una scelta tra la lettera **X** e la lettera **Y**. Analogamente, si può estendere il funzionamento a tutto un nome di file:

Delete Questo|Quello

...rimuoverà tanto un file di nome **questo** che uno di nome **quello**. Chiaramente, il simbolo trova maggiore utilità applicandolo assieme agli altri pattern, e soprattutto associandolo alle parentesi.
Vediamolo con un esempio:

Delete M(u|o)lta

...simile a quanto visto a proposito del punto interrogativo, cancellerà i files di nome **Multa** e **Molta**, ma non **malta**, come sarebbe invece accaduto adoperando il punto interrogativo.
L'uso non è certo molto frequente, ma può sempre fare comodo.

%

Percento.

Rappresenta, in associazione agli altri pattern, la presenza di un carattere **nullo**, ovvero una sua assenza.
Con il punto interogativo, ad esempio, può essere selezionata qualunque ricorrenza di un singolo carattere, ma non la sua assenza: con il sibolo *percento* si può ovviare a questo inconveniente.

Quindi, per esempio, con...

Delete MO(?|%)TO

...la ricorrenza potrà interessare **Molto, Morto, Mosto**, ma anche **Moto**. Indispensabile, in questo ma anche in quasi tutti i casi in cui si adopera il simbolo percento, l'uso delle **parentesi**, per consentire l'alternativa.
Analogamente, si potrebbe forzare la scelta ad un solo carattere o alla sua assenza. Tornando all'esempio prima visto, con...

Delete MO(L|%)TO

...l'unica alternativa possibile è rappresentata dai nomi **Molto** e **Moto**.

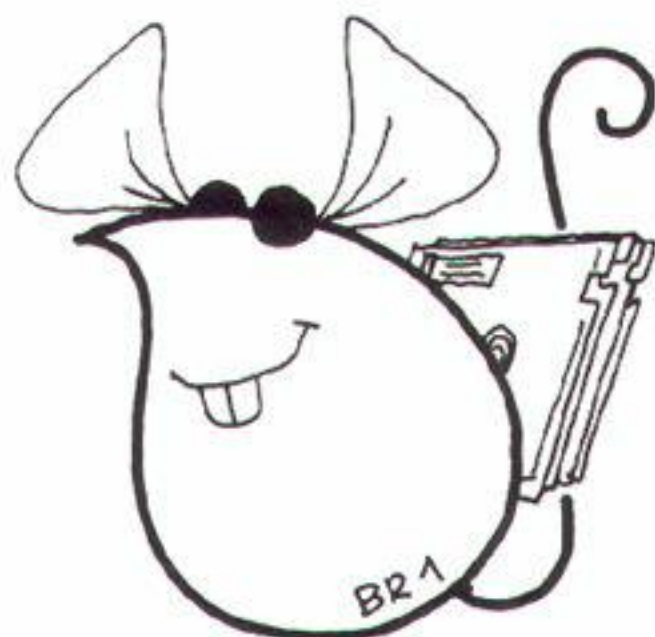
'

Apostrofo

AmigaDos consente, nel definire il nome di files o directory, l'uso di tutti i simboli, compresi quelli qui descritti per delineare dei profili di corrispondenza.
Il che, come ovvio, potrebbe creare dei problemi.
Supponiamo, ad esempio, di avere nel nostro disco un programma con nome **FileN1**, ed uno **File?1** (improbabile, ma... non si sa mai). Se volessimo eliminare quest'ultimo con un normale **Delete File?1**, provocheremmo la cancellazione anche di **FileN1**, in quanto il punto interrogativo verrebbe riconosciuto dal Dos come un pattern.
Se, però, uno dei simboli speciali è preceduto dal singolo apice, si segnalerà al sistema di interpretare come tale il carattere che segue; quindi, per evitare indesiderate rimozioni, si dovrà usare...

Delete File'?1

Comunque, anche se non obbligatorio, è buona norma non assegnare, a files o directory, nomi che possano interferire con la gestione dei pattern.

RENAME

FROM TO / AS

Keyword facoltative, con **TO** ed **AS** che possono essere adoperate indifferentemente. Come per tutte le volte che si è avuto a che fare con parole chiave facoltative, il consiglio è quello di non adoperarle, a meno che non servano a ricordare meglio le caratteristiche del comando (sempre che si abbia una certa padronanza della lingua inglese).

Nome 2

Il nuovo nome da assegnare al file. E' questo, in effetti, il parametro che decide se si sta procedendo ad un semplice **Rename**, oppure ad uno **spostamento** di directory.
In pratica, se il percorso del file è identico a quello specificato nel parametro *Nome1* (ivi compresa, come ovvio, l'assenza di un percorso), si otterrà un *Rename*.
Al contrario, una eventuale discrepanza nel percorso dei due nomi, provocherà uno spostamento fisico del file. Attenzione, però, che una possibilità non esclude l'altra.
Più chiaramente: Con Rename si può cambiare nome ad un file, modificarne la sua collocazione nell'organizzazione del disco, ma anche effettuare entrambe le operazioni contemporaneamente: spostare il file in un'altra directory, dove può assumere un nuovo nome.

In pratica:

Rename Mioprg Tuoprg

...apporterà una semplice modifica al nome del file **Mioprg**, che assumerà il nome **Tuoprg**, il tutto nell'ambito della directory corrente.
Un comando...

Rename Mioprg Sys:util/Mioprg

...trasferirà invece lo stesso file dalla directory corrente a quella **Util**, accessibile dalla directory principale (Root).

Infine, con...

Rename Mioprg Sys:Util/prog

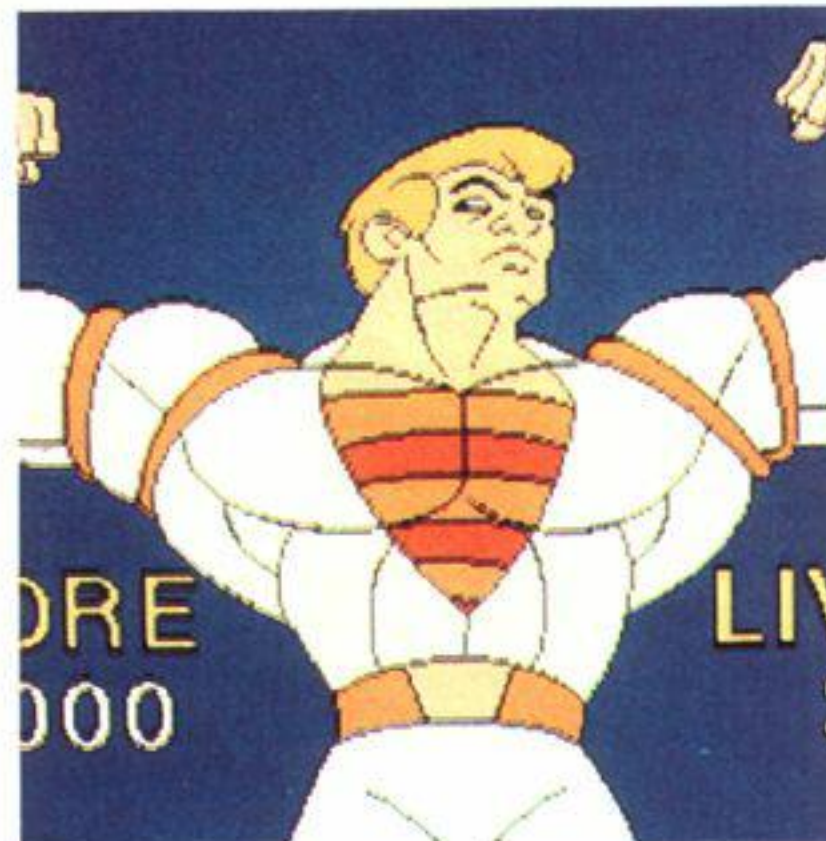
...il file **Mioprg** verrà spostato nella directory di cui sopra, dove però assumerà il nome **Prog**.

RENAME FROM nome1 TO AS nome2

RENAME

Consente di cambiare il nome di un file o di una directory, ma non quello di un disco (all'uopo, c'è **Relabel**).
L'operazione di Rename può coinvolgere l'intero percorso di un file, il che si traduce nella possibilità pratica di spostarlo fisicamente da una directory ad un'altra, anche non "contigua", purchè nell'ambito di uno stesso disco.

Nome 1

Nome del file da rinominare. Se è memorizzato nella directory corrente, si può evitare di precisarne l'intero percorso. Per un semplice **Rename**, che non comporti lo spostamento del file in un'altra directory, si presti una certa attenzione qualora si adoperi un percorso nel nome del file originario: lo stesso **path** (percorso) deve poi essere riscontrabile nel parametro successivo, o il file verrà "catapultato" altrove (vedi esempi).

ESEMPI

Rename Df1:provv Df1:def
Il file **Provv**, presente nella seconda unità a dischetti, assumerà il nome **Def**.

Rename C:Loadwb C:WB
Il comando **Loadwb**, quello che fa "partire" il Workbench, viene rinominato in un più comodo **WB**.
Si noti come il percorso specifica in questo caso il device logico **C:**, corrispondente alla directory **C** del disco di boot (salvo modifiche apportate con Assign), più comodo da usarsi che non l'intero percorso **Sys:C/Loadwb** (comunque adoperabile).

Rename :Hack/Zap Zap
Sposta il file **Zap** dalla directory **Hack** (i due punti iniziali stanno ad indicare che questa è accessibile dalla Root Directory) a quella corrente, quale che essa sia, purchè nell'ambito dello stesso dischetto.

Rename tel/mod util/modem
Sposta il file **Mod** dalla directory **Tel** a quella **Util**, ove assumerà il nome **Modem**.
In questo caso, entrambe le directory coinvolte nel comando devono essere accessibili dalla directory corrente.

Rename Prog PROG
Anche questa è una operazione di Rename, pur se di poca utilità pratica. Si limita, infatti, a modificare la rappresentazione del nome del file nei corrispondenti caratteri in maiuscolo.
Come noto, Amiga Dos non fa distinzione tra caratteri "uppercase" (maiscolo) oppure "lowercase" (minuscolo), ma anche l'occhio può pretendere la sua proverbiale parte...

ATTENZIONE A...

L'unico rischio legato a Rename, è quello di operare uno spostamento non voluto, cosa che può creare l'illusione di una involontaria cancellazione. In questi casi, basta impartire un **Dir Opt A** per scandire tutto il contenuto del dischetto, oppure sfruttare il più evoluto **Which Nomefile**, che visualizzerà il nuovo percorso del file.
Altra possibilià, alquanto remota, è quella che si sia assegnato al nuovo file un nome composto di soli spazi. La cosa, però, difficlmente può essere involontaria: occorrerebbe infatti impartire **Rename Mioprg " "**, con le virgolette.
Il problema, comunque, non riguarda Rename, che svolgerebbe il suo compito come sempre: solo che il nome del file non risulterebbe più visibile, e per lanciarlo si renderebbe necessario digitare le virgolette, lo spazio, ed ancora virgolette. Se, poi, si fosse adoperato più di uno spazio, l'unico problema sarebbe il ricordarsi quanti sono, al momento di eseguire il programma: neanche un List (Dir, peggio ancora) sarebbe più d'aiuto.

ERRORI PIU' FREQUENTI

Il comando Rename, nel precisare l'errore in cui si è incorso, è alquanto laconico.

Escludendo il solito **Bad Args** (errore nella sintassi), una segnalazione ricorrente è **"Can't rename..."**, che può comprendere le più svariate situazioni.

Nella maggior parte dei casi si tratta di una non corretta specifica di percorso, che rende inaccessibile il file originario al comando Rename. Ma può anche segnalare (si fa per dire...) che si è tentato di spostare un file in unità - disco diverse (Ram: compresa!).

L'errore può anche verificarsi nel caso esista già un file con lo stesso nome di quello che si intende assegnare al file origine. In questo caso, a differenza di comandi come Copy, il file **non** viene sovrascritto, e la procedura di Rename si interrompe.

Forse una segnalazione della cosa, con relativo requester di scelta, sarebbe risultata più comoda, ma Rename (quello fornito dalla Commodore, quanto meno) è fatto così.

THE WORKS!

(a cura di Goran Fisher)

The Works! è un cosiddetto **pacchetto integrato** prodotto dalla Micro Systems Software (12798 W. Forest Hill Boulevard, Suite 202, West Palm Beach, Florida, USA) comprendente 4 programmi: **Word Processor, Spreadsheet,** programma di **Telecomunicazione** e **Database**. Si dice **pacchetto** perchè consiste di più programmi separati, **integrato** perchè i programmi che lo compongono sono in grado di scambiarsi i dati.

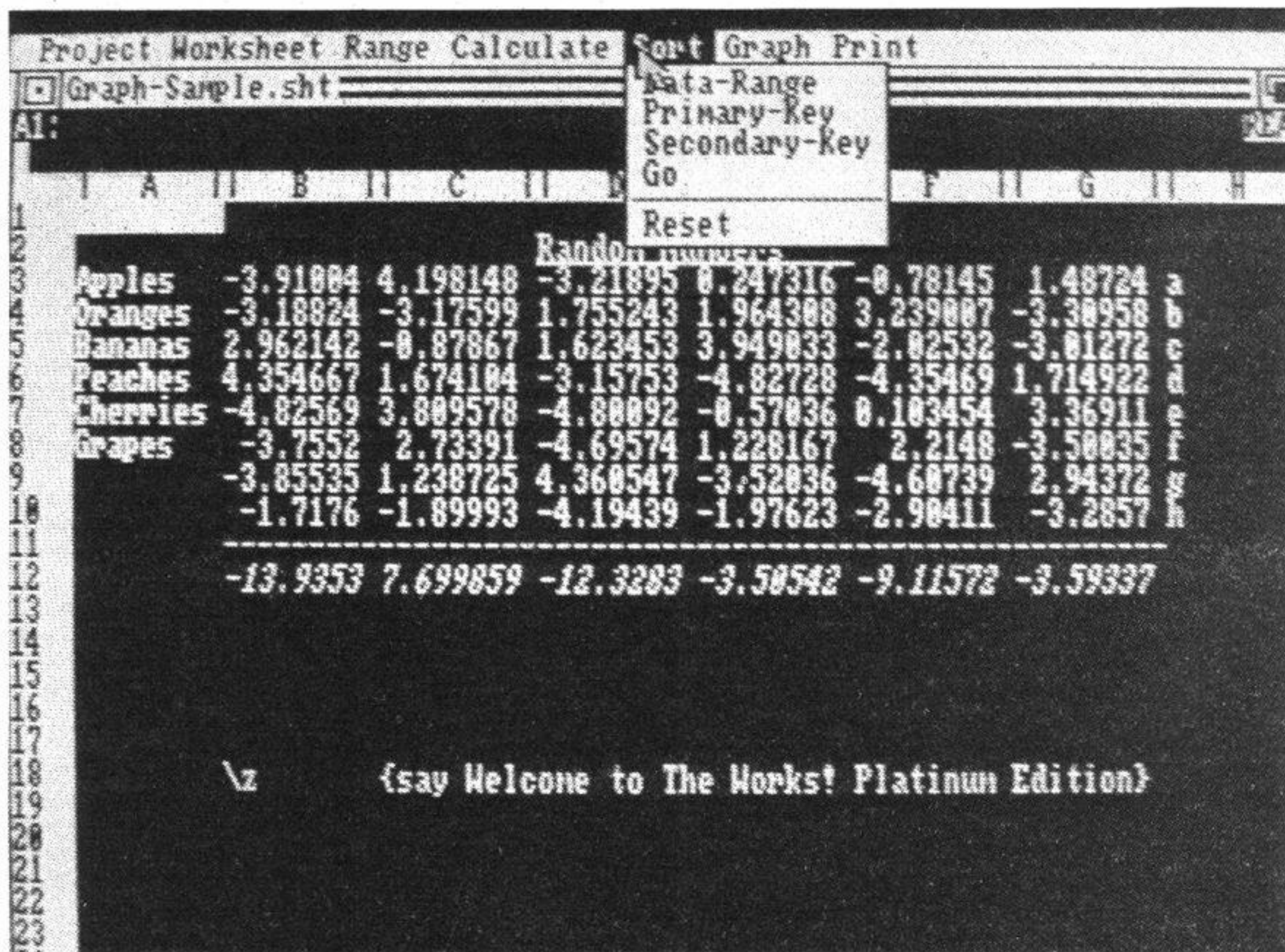
Ad esempio, un file creato con il programma di videoscrittura (che è **Scribble!** Platinum, in effetti) può essere direttamente "passato" al sistema di archiviazione, oppure essere usato come base di dati del tabellone elettronico.

La caratteristica dei programmi che compongono The Works!, comunque, è l'estrema semplicità d'uso, che li rende adatti ad essere usati da tutti, in particolare da chi non ne fa un uso superevoluto. Trattandosi di quattro programmi distinti, la descrizione delle funzioni disponibili occuperà più numeri di CCC, per evidenti motivi di spazio.

Inizialmente si dive clickare nell'icona a forma di mano del disco di Workbench (i dischi di The Works! sono **tre**), che lancia un programma supervisore in grado di richiamare gli altri moduli, i quali gli restituiscono il comando quando ne viene terminata l'esecuzione.

IL WORDPROCESSOR

Il programma di videoscrittura integrato in The Works! è il caro e vecchio Scribble!, nella sua più recente riedizione, quella chiamata **Platinum**. Si tratta di un Word Processor relativamente semplice, ma molto potente e completo di tutte le funzioni e caratteristiche di base di programmi molto più potenti e costosi.

Diamo subito una nota d'uso, che molti non conoscono, circa i requester dei programmi di The Works!. Quando sul video si ha un requester di qualunque tipo con dei gadget contenenti stringhe (Open, Drive, Sort...) si può rispondere sia clickando normalmente col mouse su tali gadget, sia premendo su tastiera la lettera corrispondente: ad esempio **O** per Open, **S** per Select eccetera.

Menu project

Questo menu controlla le funzioni associate al documento nella sua totalità e alle operazioni di lettura e scrittura dalla memoria di massa (floppy disk, hard disk) dei documenti.

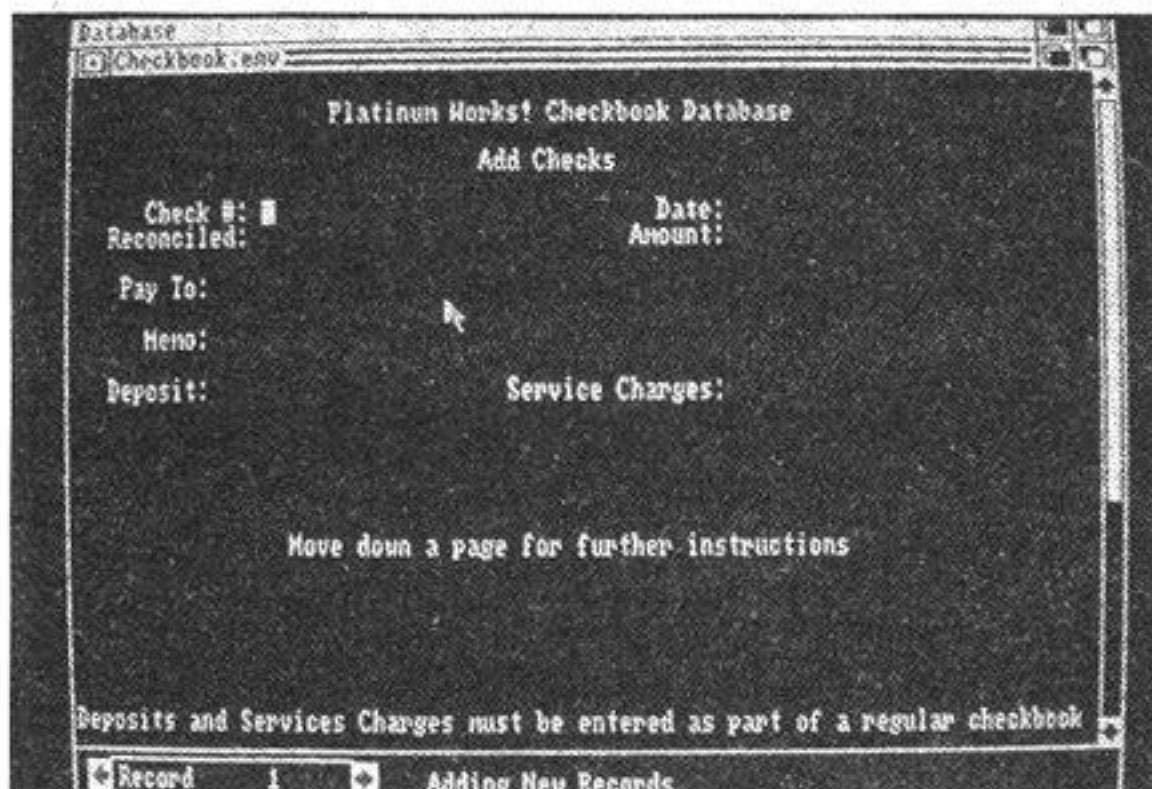
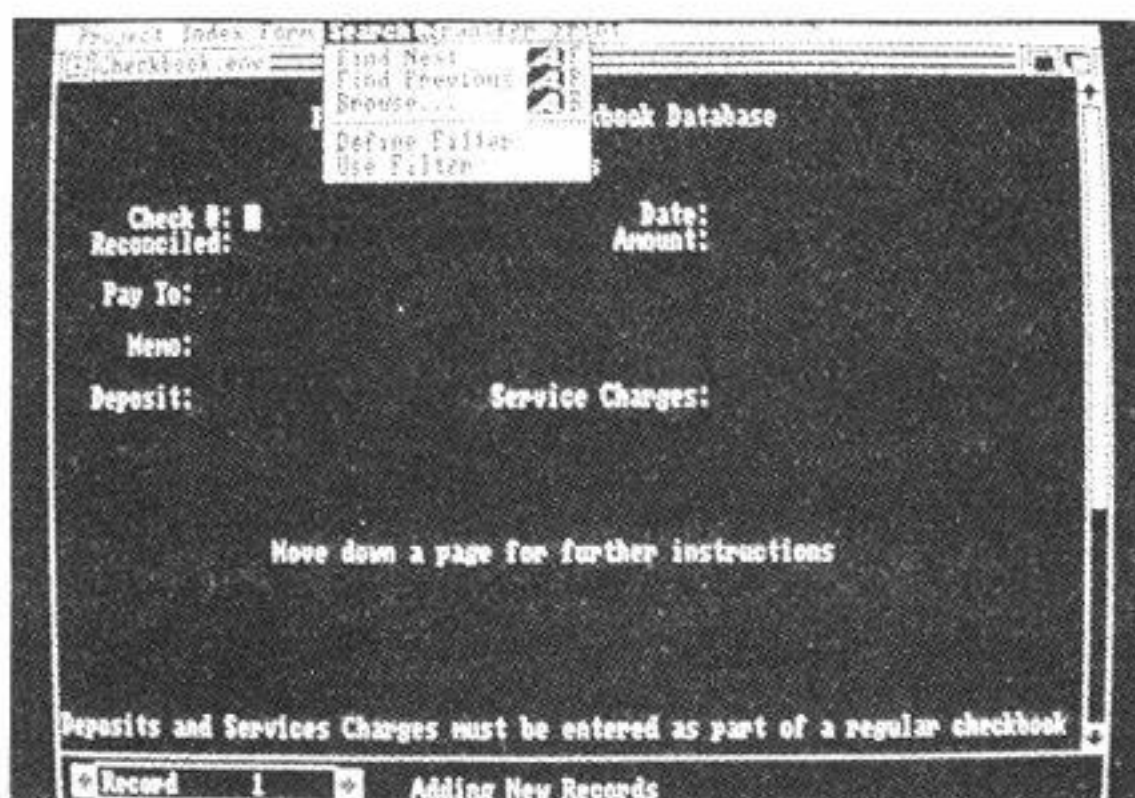
New. Apre una finestra vuota sul documento. Avendo memoria RAM a sufficienza è possibile aprire sino a *quattro* finestre contemporaneamente.

Open. Consente di caricare documenti. Fa comparire un requester di file con l'elenco delle directory e dei files dalla directory corrente. Si noti che vengono visualizzati soltanto i files col suffisso *.doc*, ovvero quelli normalmente generati dal WP stesso. Se si desiderano vedere tutti i files della directory, indipendentemente dai suffissi, si deve clickare sulla scritta accanto a *Pattern* col mouse, commutandola tra *.doc* e *.all*. Per caricare un file si può clickare due volte sul suo nome oppure digitarlo sul gadget di stringa posto accanto a *Selection* e poi clickare sul gadget *Open*.

Close. Chiude la finestra di redazione correntemente attiva, cioè quella dove si sta scrivendo, ovvero quella con il nome nella barra di intestazione scritto chiaramente e non *ghosted* (in italiano si tradurrebbe *fantasmato*, ma non è molto bello...) dal sistema operativo.

Se il documento non è stato ancora salvato dopo l'ultima modifica apportata, prima di chiudere la sua finestra verrà chiesto se si desidera effettuare l'aggiornamento del file su disco, altrimenti le ultime modifiche verranno perdute. Se la finestra è una sola, si ottiene l'uscita dal programma ed il rientro nel menu principale di gestione di The Works!

Save. Le funzioni *Save* e *Save As* sono molto simili tra loro.

La differenza è che la prima salva il documento corrente usando l'ultimo nome specificato; la seconda fa comparire un requester simile a quello di caricamento. Si noti che il programma conserva la cosiddetta *path* di salvataggio di ogni file, ovvero le directory e sub directory dove scriverlo, nel gadget stringa inferiore nella finestra di I/O dei files, che è quindi indipendente da quello superiore dove si specifica il volume (disco).

Info. Fornisce semplici informazioni sul programma: lunghezza della pagina fisica (per default: 6 linee per pollice, 66 linee per pagina), spaziatura dal margine sinistro in colonne, linee vuote lasciate come margine superiore, linee vuote lasciate come margine inferiore, lunghezza della linea, tipo di giustificazione usata a destra *(ragged o flush)*, numero di caratteri presenti, numero di caratteri ancora liberi nel buffer, numero di parole, numero di pagine.

Works. Attiva la barra di menu del programma supervisore di Platinum Works!, consentendo così di lanciare (memoria permettendo) un'altra applicazione in multitasking.

Quit. Esce dalla videoscrittura e rientra nello schermo principale di The Works!. La combinazione di tasti è Amiga destro (d'ora in poi, semplicemente *Amiga D*) e *Q*. Compare comunque un requester che chiede conferma dell'operazione tramite due gadget, ai quali si può rispondere via mouse o tastiera.

Menu preferences

Questo menu controlla alcune funzioni ed azioni del word processor, come ad esempio il tipo di giustifica, i colori dello schermo ed altro. I dati possono essere salvati e ricaricati (automaticamente all'avviamento del programma, o manualmente tramite requester) come files speciali detti *di formato*, che dispongono del suffisso *.fmt* nel nome. Ciò consente, ad esempio, di configurare una volta il programma per scrivere lettere ed un'altra volta per elaborare testi sorgente: sarà sufficiente qualche istante per caricare e "regolare" il Word Processor. Il formato di default, ovvero quello letto automaticamente dal programma quando lo si avvia, è letto dal file *Word.fmt*.

Load. Carica un file di formato. Appare un requester che elenca le directory ed i files terminanti con *.fmt*, come per il requester di carica dei documenti (che elenca quelli terminanti con *.doc*).

Save. Fa comparire un requester di registrazione su disco che consente di specificare il nominativo con il quale verrà salvato il file di configurazione. La lista ha gli elementi *ghosted* perchè siamo in salvataggio, quindi non è possibile selezionare un nome da qui. Se si digita un nome di file già presente, viene chiesta conferma prima di eseguire la sua sovrascrittura con i parametri attuali.

Editing (Mode Overtype, Insert). Commuta il modo di inserimento del testo tra "sovrascrittura" ed "inserimento". Nel primo caso quanto viene digitato dalla tastiera viene scritto nella posizio-

ne del cursore sopra caratteri eventualmente esistenti, esattamente come una macchina da scrivere, nel secondo caso i caratteri vengono inseriti alla posizione del cursore facendo scorrere verso destra la porzione di testo già presente.

Justification (Ragged, Flush). La *giustificazione* è la caratteristica dei word processor di allineare le colonne a destra. In pratica il programma inserisce, nelle linee, delle spaziature tra le parole per garantire che ogni linea sia effettivamente lunga sempre il numero massimo di colonne (vedi opzione *Preferences / Line Length*), ottenendo così un testo più ordinato. Questa è chiamata giustificazione *flush*, mentre il modo *ragged* lascia le linee così come vengono digitate. Si noti che per ottenere la giustificazione vengono inseriti i cosiddetti *soft spaces*, ovvero dei caratteri di spaziatura interni, che seppure sono visualizzati e stampati come tali, possono però essere visionati premendo *Shift-F10*, che li colora in grigio. Bisogna usare cautela quando si ha la giustificazione di tipo Flush inserita ed il modo Overtype, in quanto, sovrascrivendo un soft space con una spaziatura effettiva, la linea potrebbe non venire più giustificata correttamente dal programma. Infatti il word processor provvede ad inserire e cancellare i soft spaces automaticamente, di propria iniziativa, durante l'inserimento stesso del testo, per garantire l'allineamento, mentre non può prendersi la stessa libertà di gestione con uno spazio tradizionale. Si noti, inoltre, che quanto viene stampato può differire da quanto si vede sullo schermo se si sono usati i *dot commands* nel testo (visibili premendo F2), in particolare *.JU* e *.FR*, che sono però visibili nella Overview ottenibile dal programma (menu Print).

Line Length. Consente di fissare la lunghezza della linea affinchè concordi con la lunghezza della linea stampata effettivamente su carta, in virtù del tipo di fonte usata dalla stampante. Una finestra standard può presentare tipicamente *76 caratteri* visibili su ogni linea; eccedendo, si ottiene uno scrolling orizzontale automatico da parte del programma. Come sempre, il valore qui specificato

viene usato per l'intero testo con l'eccezione dei punti dove si sono eventualmente inseriti dot commands *(.LL .PO .LM .RM).* Il testo viene comunque sempre centrato alla larghezza dello schermo se si sceglie una larghezza di linea inferiore ai 76 caratteri.

Tabs. Definisce la posizione dei tabulatori. Un *Tab Stop* è un punto dove si

arresta il cursore automaticamente quando si preme il tasto tabulatore. Per specificare tali posizioni, si devono immettere, nel requester che compare, i numeri di colonne consecutivi separati da virgole, clickando su *OK* alla fine o su *Cancel* per annullare.

Foreground. Consente di stabilire i colori del primo piano (il corpo dei caratteri) del testo, tramite apposito requester. Si noti che i colori sono selezionabili tramite la funzione *Project / Preferences* dello schermo di lavoro principale. Ciascuna finestra attiva può avere una tavolozza cromatica differente, essendo effettivamente uno Screen di Intuition.

Background. Funziona in modo analogo a Foreground, ma modifica il colore dello sfondo.

Markers (Hide, Show). Commutano la visualizzazione sullo schermo dei marcatori di paragrafo, come la combinazione SHift-F9.

Print **Setup Page Length.** Fissa l'altezza fisica della pagina, ovvero il numero di linee che verranno prodotte per ciascuna delle pagine del documento scritto. Per default vengono usare *66* linee, adatte per il formato di carta americano.

Da noi, usando i fogli di formato *A4*, è consigliabile usare *59 / 60* linee. Questa regolazione può essere ottenuta anche col dot command *.PL.*

Print **Setup Page Offset.** Numero degli spazi che vengono effettivamente stampati a sinistra prima del testo vero e proprio, per centrare in pratica il testo sul foglio.

Eventualmente si può stabilire un valore differente per le pagine pari e dispari, ideale quando si stampa in doppia facciata.

Equivale ai comandi *.PO, .EO* e *.OO* per offset di pagina, offset di pagine pari ed offset di pagine dispari.

Print **Setup Top Margin.** Stabilisce il numero di linee lasciate vuote durante la stampa in testa alla pagina, con un default di 6. Equivale al comando *.MT.*

Print **Setup Bottom Margin.** Stabilisce il numero di linee lasciate vuote durante la stampa in fondo alla pagina, con un default di 6. Equivale al comando *.MB.* Si noti che il numero di linee effettivamente usabili può così essere calcolato come: lunghezza pagina - (margine superiore + margine inferiore), in quando anche le linee lasciate in testa ed in fondo alla pagina rientrano

nel conteggio del numero di linee per pagina (66 per default).

Print **Setup Special**. Visualizza un requester dove si possono digitare dei codici di controllo che vengono inviati direttamente via la configurazione scelta tramite il programma Preferences del Workbench, oppure tramite *SER:* o *PAR:*. Ciò dipende dalla selezione del gadget *Direct:* se è evidenziato, viene saltata l'interfaccia predisposta da Preferences.

File Format (Lf Only; Cr+lf; Cr only). Con queste opzioni si controlla il formato effettivamente utilizzato dal programma per marcare la fine della linea. Normalmente, secondo lo standard Amiga, è sufficiente usare il codice di *Line Feed*, ma ad esempio per inviare files ad una BBS si deve tipicamente usare le combinazione *Cr+lf*. Il CR di chiusura è stato inserito per compatibilità con vecchi formati di files.

Menu mode

Le sue funzioni controllano le operazioni che agiscono sulla globalità del testo oppure su porzioni definite, generalmente chiamate *blocchi*.

Edit. Il modo di redazione è normalmente il più comune in un word processor. Quando è attivo, il puntatore del mouse ha la sagoma di una matita, e consente di fare scorrere il testo o posizionare istantaneamente il cursore.

Cut. Questo modo, simile al *Copy* (vedi dopo), consente di eliminare una porzione di testo con una sola operazione. La sagoma del puntatore del mouse assume la forma di una una forbicina. Clickando in un punto e mantenendo premuto il pulsante sinistro, si evidenzia (colore diverso) una parte di testo. Quando si rilascia il pulsante, la parte evidenziata viene cancellata. Questo modo consente anche di spostare un blocco, usando subito dopo la funzione *Paste* (vedi sotto). Si noti che il blocco tagliato non viene perduto, ma conservato in un apposito buffer *(Clipboard buffer)* pronto per essere eventualmente recuperato con la funzione *Paste*.

Copy. Il comando Mode / Copy lavora come Cut, con la differenza che non elimina affatto il testo sullo schermo, ma lo immagazzina nel clipboard. Con esso è possibile, ad esempio, trasferire blocchi tra le finestre.

Paste. Il comando inserisce alla posizione del cursore quanto presente nel clipboard. Qui risiede normalmente una porzione di testo depositata dalle operazioni Cut o Copy. L'operazione non cancella il contenuto del clipboard, quindi può essere ripetuta più volte.

Style (Plain, Bold, Underline, Italic, Superscript, Subscript). Come intuibile da chiunque abbia anche soltanto un'infarinatura di word processing, queste funzioni fissano il *modo* di scrittura: normale, **grassetto**, sottolineato, *corsivo*, apici, pedici. Alcuni sono anche replicati nei tasti funzione F6 (bold), F7 (underline) e F8 (italic) per un più rapido uso da tastiera. Gli stili possono essere mescolati a piacere, ma possono dare origine anche a stringhe difficilmente leggibili sul monitor. Inoltre, non tutte le stampanti supportano tutti questi stili e non tutte le fonti di caratteri delle stampanti prevedono di mescolarli liberamente.

Menu document

Il menu Document controlla alcune caratteristiche speciali, principalmente le funzioni di ricerca e sostituzione di parole nel testo. Lo *Spelling Checker* ed il *Theasaurus* sono di scarsa utilità per gli utenti italiani.

Find. Consente di localizzare un carattere, o preferibilmente una stringa, nel testo. Si deve digitare, nel requester che compare, la stringa desiderata e premere il tasto Return per iniziare la ricerca. Questa parte sempre dall'inizio del documento e procede verso il basso.

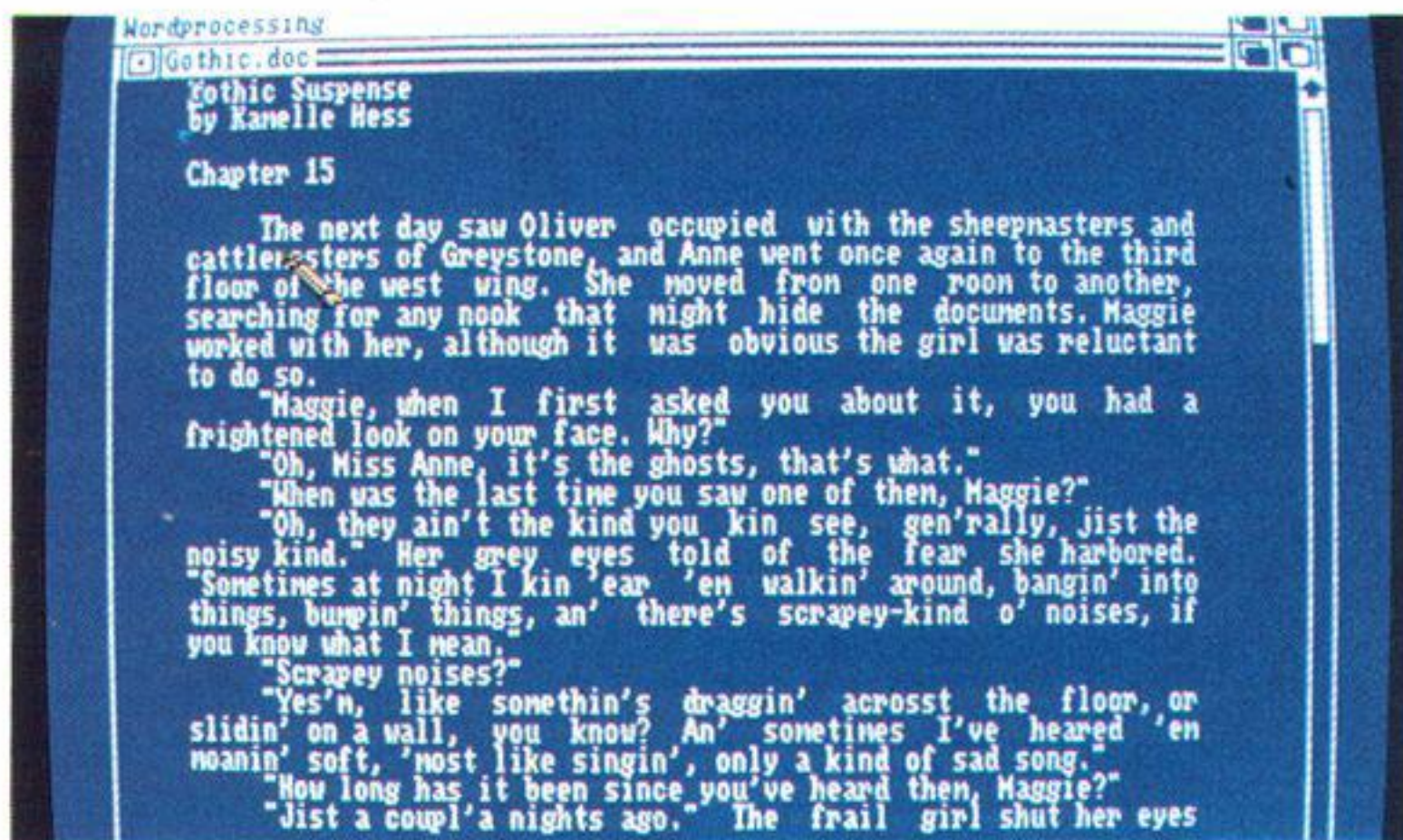
Con i tasti *AmigaS + A* si può ripetere la ricerca, mentre la sequenza *AmigaD + F* fa comparire in qualunque momento il requester di ricerca. La stringa cercata rimane nel buffer e si può quindi ripetere la ricerca successivamente con molta rapidità.

Replace. Questa funzione lavora in stretta congiunzione con Find, consentendo di sostituire una certa parola ad un'altra ricercata. Infatti, Replace funziona soltanto dopo che si è localizzata la stringa cercata. In questo caso compare un requester con quattro opzioni: *Skip, Change, All* e *Quit*.

Con la prima si annulla l'operazione di sostituzione della stringa localizzata con quella specificata nel requester per la sua sostituzione e si procede alla ricerca della successiva. Con *Change* si effettua la sostituzione e si prosegue nella ricerca. Con *All* si effettuano tutte le sostituzioni automaticamente nel testo senza chiedere conferme, mentre con *Quit* si annulla l'operazione di Find/Replace.

Spell. Attiva lo Spelling Checherk. Purtroppo il programma prevede soltanto la lingua inglese, quindi è utile solo per correggere documenti in questa lingua.

Spell. *Guess:* verifica lo spelling di una sola parola, quella evidenziata col mouse. *Document:* verifica lo spelling dell'intero documento. *Window:* verifica lo spelling della sola porzione di testo visibile.

Continuously: commuta il modo di verifica spelling durante la battitura. *Thesaurus:* controlla il vocabolario inglese interno al programma.

Menu print

Come prevedibile dal nome, questo menu controlla le operazioni di stampa su carta.

Preview. Esegue una *stampa* a video del documento. Ciò significa che vedremo effettivamente il documento come dovrebbe apparire sulla periferica, compresi tabulatori, margini, intestazioni, note, eccetera, in quanto vengono letti ed eseguiti tutti i comandi di formattazione ed i dot commands.

La preview inizia sempre dalla posizione corrente del cursore, quindi si consiglia di collocarlo sul primo carattere del documento per avere una visione precisa e corretta. Premendo la *barra spaziatrice* durante la visione si ottiene una pausa, mentre premendo *Esc* si annulla la preview. Gli unici dot commands non rispettati, per ovvi motivi, sono il page offset *(.PO)* e insert graphic *(.IP)*.

Forward. Consente di fare partire la stampa su carta dalla linea sulla quale è collocato attualmente il cursore. E' consigliabile inserirlo, quindi, nella prima linea di una pagina, per ottenere una stampa corretta.

Go (Printer, File). Fa partire materialmente la stampa del documento attuale. Con *Go Printer* il documento viene inviato a *PRT:*, quindi stampato con l'interfaccia software di AmigaDOS, come configurata dal programma Preferences del Workbench. Con *Go File* l'output del programma viene invece inviato ad un file, di cui si deve specificare il nome secondo lo standard AmigaDOS (ad esempio, *df0:Provastampa)*, che quindi alla fine conterrà tutto il testo con anche i codici ASCII di controllo per tabulatori, stili, margini eccetera.

Alla fine, si può anche inviare da Shell tale file alla stampante con una linea del tipo COPY df0:Provastampa TO PRT: ottenendo lo stesso effetto di un Go Printer diretto.

Per sospendere momentaneamente una stampa si può usare la barra spaziatrice, mentre con Esc si annulla l'operazione.

Quality (Draft, NLQ). Le funzioni consentono di commutare tra stampa ad alta velocità *(draft)* e stampa ad alta qualità *(Near Letter Quality)*. Ovviamente funzionano soltanto se la stampante connessa prevede effettivamente un modo draft ed un modo NLQ.

Paper (Type Fanfold; Type Single). Consentono di indicare al programma se nella stampante abbiamo inserito carta continua *(fanfold)* oppure fogli singoli *(single)*. Nel primo caso, la stampa su carta proseguirà senza interruzioni, mentre nel caso si indichi l'uso di fogli singoli, il programma sospenderà l'invio alla periferica alla fine di ogni pagina per consentire la sostituzione del foglio singolo, per riavviarsi alla pressione di un tasto.

Page Number. Dice al programma di videoscrittura di iniziare la numerazione delle pagine per il documento corrente con un qualunque valore, anche differente da uno. Ciò consente, ad esempio, di stampare un solo documento, ad esempio un libro, materialmente suddiviso in più files (per ragioni di efficienza o di memoria) consecutivamente, senza sbagliare i numeri di pagina.

Line Spacing. Consente di specificare la spaziatura delle linee usata dal WP. Normalmente si usa 1, ad indicare che la spaziatura delle linee è singola, quindi direttamente dipendente dal parametro "linee per pollice" usato dalla stampante (e fissato dal programma preferences del Workbench). Il valore usato per Line Spacing viene usato per tutto il documento, a meno che non si specifichi una linea di formattazione differente.

Copies. Consente di specificare il numero di copie da stampare, che per default è una sola. Questo comando, comunque, non fa partire la stampa.

TELECOMUNICAZIONE

Il programma di telecomunicazione presente in The Works! è una versione perfezionata di **OnLine!**, con alcune aggiunte e modifiche marginali.

Si tratta di un programma molto evoluto, comprendente tra l'altro il protocollo **Z-Modem** per il trasferimento di files, la possibilità di gestire files script, vari modi di configurazione, otto colori e molto altro.

Menu project

Questo menu è simmetrico al precedente per il WordProcessor, consentendo le operazioni di I/O dei files e la configurazione del programma di telecomunicazione.

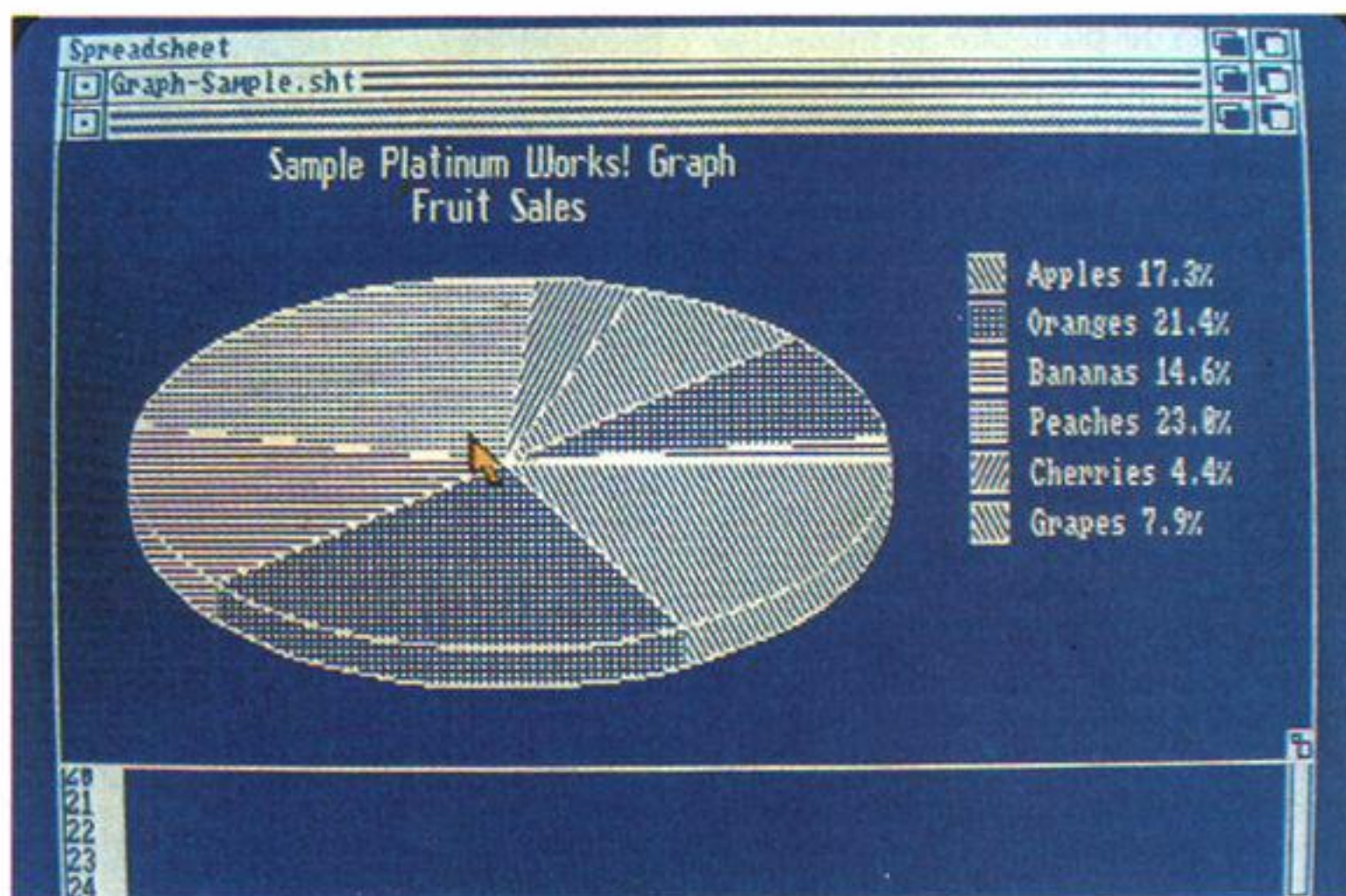
New. Consente di aprire la finestra del terminale.

Open. Carica un file di archivio o di configurazione del programma.

Close. Scrive i cambiamenti apportati sul file di terminale.

Save (As). Memorizzano permanentemente i files script, di configurazione o di elenchi telefonici. Il primo comando salva per default con il nome specificato precedentemente con il secondo comando.

Phone Book. Fa comparire il requester contenente sino a 40 numeri telefonici e formati di chiamata.

Call. Consente di specificare direttamente un numero da chiamare.

Hang Up. Interrompe il collegamento con il terminale remoto inviandogli anche un apposito messaggio.

Clear Screen. Elimina dallo schermo tutti i messaggi presenti.

Review. Fa rivedere gli ultimi 8K ricevuti dal terminale in una finestra. Si accede anche ad un sottomenu che consente di scambiare parti del file con gli altri programmi di The Works tramite evidenziazione col mouse.

Printer. Visualizza un menu pop-out per attivare e disattivare l'invio dei messaggi alla stampante.

Info. Visualizza informazioni e statistiche di lavoro.

Works. Attiva la barra del programma supervisore consentendo di chiamare in multitasking le altre applicazioni.

Quit. Termina l'esecuzione del programma di telecomunicazione.

Menu preferences

Qui sono contenute le funzioni di configurazione del programma.

Modem. Fa comparire un requester con i parametri da usare per il modem.

Chat. Attiva il modo di *chatting*, visualizzando una finestra in cui si possono scrivere i messaggi che verranno inviati al terminale remoto (che risponde usando altra finestra). Per uscire si clicka sul gadget di chiusura della finestra inferiore.

Font. Consente di scegliere la fonte di caratteri da usare nel programma: 5 x 8, 8 x 8, 10 x 9 oppure 8 x 11.

Leading. Aumenta la distanza tra le linee. Particolarmente utile quando si usa uno schermo interlacciato.

Title. Visualizza un menu pop-out che attiva e disattiva la barra del menu.

Border. Visualizza un menu di pop-up che attiva e disattiva il bordo della finestra.

Foreground. Colora i caratteri.

Background. Colora lo sfondo.

Clock. Consente di scegliere se visualizzare l'ora corrente, il tempo trascorso dall'inizio del collegamento, oppure "spegnere" l'orologio nella barra del menu.

Beep. Attiva o disattiva i messaggi sonori.

Menu file

Questo menu contiene le funzioni di gestione dei files.

Receive. Consente di eseguire il trasferimento in un file su disco di un pacchetto inviato dal terminale remoto quando si è scelto il *downloading*. Bisogna specificare il nome del file di destinazione dopo la scelta del formato (X-Modem, Kermit...)

Send. Consente di inviare un file al terminale remoto, usando il formato di pacchetto voluto (X-Modem...). Bisogna ovviamente indicare il nome del file in formato Amigados.

Path. Visualizza la path del drive di default per i files ricevuti.

Protocol. Mostra un menu pop-out che elenca i protocolli di trasferimento previsti. Si deve scegliere uno tra quelli previsti nella BBS che si collega. Ad esempio per la **BBS Systems** va molto bene X-Modem.

Type. Consente di indicare al programma se si sta trasferendo un file di testo o binario.

Auto Chop. Attiva o disattiva il modo di eliminazione automatica dei bytes extra posti in fondo al file da molti protocolli, quando è necessario un arrotondamento a numero pari dei blocchi.

EOL Conversion. Consente di scegliere se convertire automaticamente i fine linea in codici LF normali, in CR od in LF + CR. Normalmente si usa il primo.

Icon. Consente di scegliere se si vuole automaticamente una icona per l'applicazione salvata come file o meno.

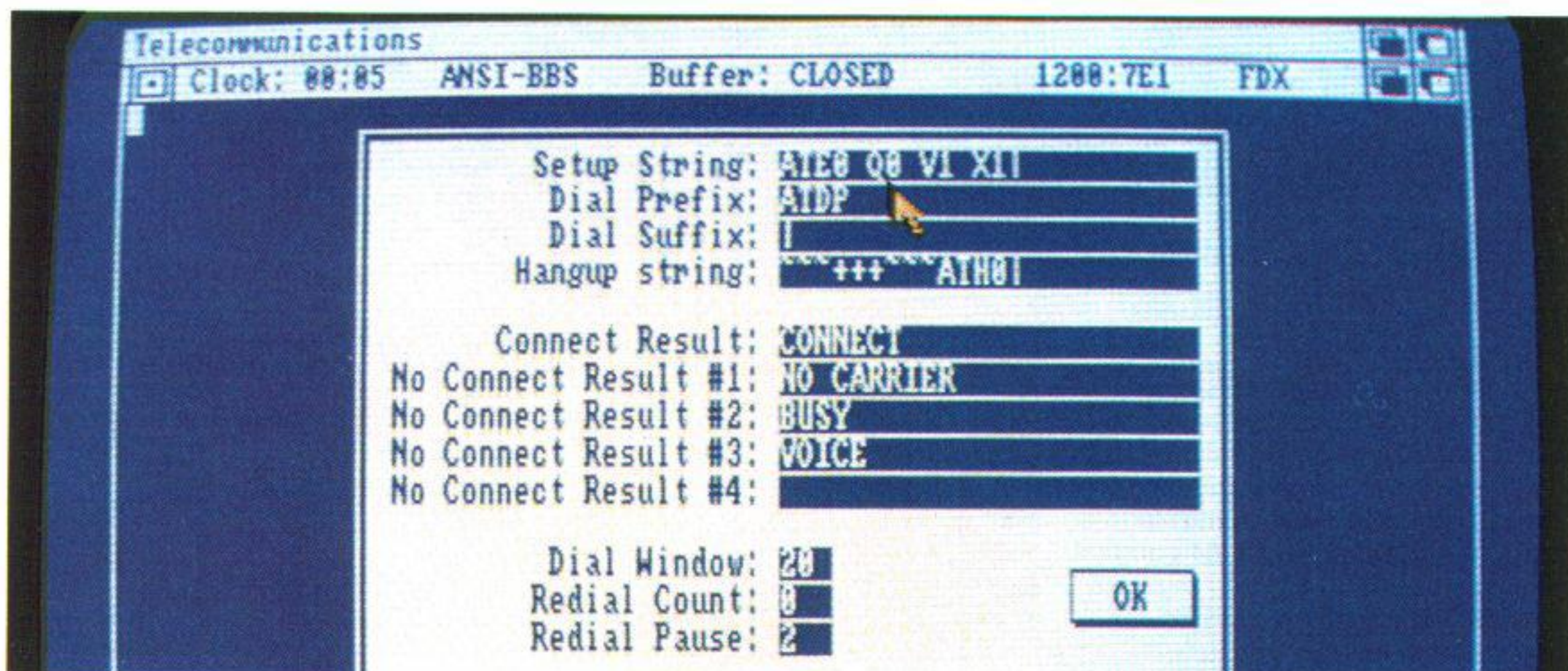
Menu buffer

Questo menu contiene tutte le opzioni inerenti al buffer di conservazione del testo *(capture buffer)*. Tale buffer consente di catturare il testo visualizzato nella finestra in un file specificato, oppure ad un terminale remoto in un secondo tempo.

Load. Carica il file nel buffer di cattura, dopo avere specificato il suo nome.

Save. Memorizza permanentemente i files su disco, dopo avere specificato il nome standard Amigados desiderato.

View. Mostra una finestra con i contenuti del capture buffer.

Open. Apre fisicamente in memoria il buffer, che per default vale 16K, ma si può specificare un valore differente,

compatibilmente con le capacità del proprio Amiga.

Close. Chiude il buffer aperto col comando precedente, salvandone i contenuti nel file indicato.

Send. Invia il testo contenuto nel buffer di cattura ad un sistema remoto.

Send **Stop**. Sospende l'output del buffer al terminale remoto.

Send **Go**. Inizia l'invio del buffer al terminale remoto.

Send **C-Delay**. Inserisce un ritardo per ciascun carattere inviato. Il buffer di cattura viene usato appropriatamente dai sistemi che prevedono effettivamente l'handshake *Xon / Xoff*.

Il valore di ritardo è da specificare in millisecondi, con un numero compreso tra 0 e 255.

Send Prompt. Il programma di telecomunicazioni può essere configurato per inviare una linea dal buffer di cattura soltanto quando viene presentata una specifica richiesta dal sistema remoto.

Menu script

Questo menu mostra le opzioni connesse all'esecuzione ed alla scrittura dei files *script*, che consentono al programma di eseguire una serie di comandi letti da un file esattamente come se fossero impartiti direttamente dal programmatore.

Load. Legge il file, di cui si specifica il nome, come uno script da eseguire.

Save. Salva lo script contenuto nel relativo buffer.

View. Mostra una finestra che visualizza i contenuti del buffer di script.

Script Stop. Ferma l'esecuzione di uno script.

Script **Go**. Esegue lo script dall'inizio.

Script **Resume**. Continua l'esecuzione dello script dopo aver reimpartito un comando di Stop.

Script **Clear**. Cancella lo script.

Menu setup

Questo menu contiene le funzioni che consentono di configurare i tasti di macro, i tipi di terminale ed altri setting.

MKeys. Ridefinisce i 10 tasti funzione ed i 10 tasti di Shift, che consentono al terminale di inviare sino a 64 caratteri con un solo tasto.

Width. Riformatta il display del terminale alla larghezza specificata.

Tables. Consente di scegliere una tra sette tabelle di traduzione, che consentono ai programmi di telecomunicazione di convertire automaticamente dei bytes. Ciò avviene allo scopo di filtrare od alterare caratteri dannosi.

Tables Display. Altera i caratteri prima che vengano visualizzati nella finestra del terminale. Ad esempio, cambiando il codice *08* (BS) in *7F* (DEL) si ottiene un effetto distruttivo della pressione del tasto di backspace.

Tables Printer. Altera dei caratteri prima che vengano inviati alla stampante, quando attivata.

Tables Keyboard. Converte i codici che arrivano dalla tastiera.

Tables **from Buffer**. Altera i caratteri letti dal capture buffer.

Tables **to Buffer**. Altera i caratteri inviati al capture buffer.

Tables **in Serial**. Converte i caratteri ricevuti dal computer via RS232.

Tables **out Serial**. Altera i caratteri inviati dal computer via RS232.

Emulation. Sceglie il tipo di emulatore di terminale: VT100, VT102, VT52, TEK-4010, TTY. Per la BBS Systems va bene VT100.

Duplex. Consente di indicare al programma se il modem è in grado di dialogare o solo di ricevere o spedire. Il primo caso è quello normale, e richiede di specificare Full Duplex.

Echo. Attiva o disattiva l'echo. Deve essere attivato solo quando almeno uno dei due terminali è in half duplex.

CR + LF. Consente di scegliere se tradurre le sequenze di Carriage Return e Line Feed usati nei files. Se attivata, la conversione traduce il LF in CR + LF in trasmissione ed esegue la procedura contraria in ricezione.

Flow. Attiva o disattiva il flusso dei dati, tramite le linee *Xon / Xoff*, se utilizzate.

Con *Hardware* vengono usati i piedini 5 e 6 della RS232, con *None* si ignorano le linee di handshaking mentre il default *Software* è il più usato.

Com menu

Qui sono contenute le funzioni di controllo del device di comunicazione.

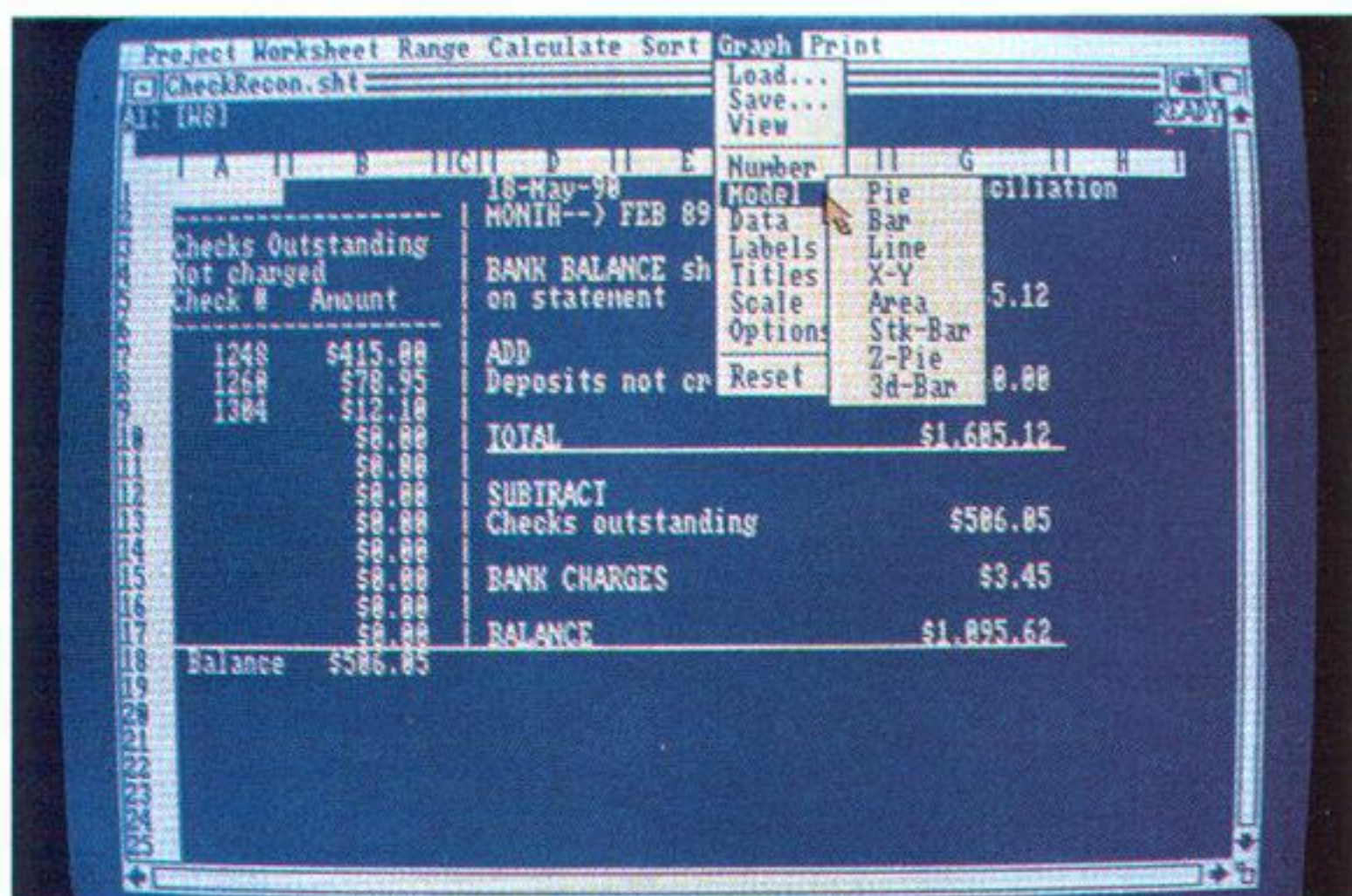
Baud. Regola la velocità di trasmissione e ricezione dei dati. Per **BBS Systems** vanno bene i valori 300, 600, 1200.

Ovviamente il nostro modem deve prevedere queste velocità.

Word. Consente di indicare quanti bit si usano per un carattere: 7 oppure 8. Per la nostra BBS se ne usano 7.

Parity. Consente di scegliere il tipo di controllo di parità eseguito dal programma sui caratteri: nessuno, pari, dispari, mark, spazio. Per la nostra BBS si deve usare **E**.

Stops. Consente di indicare quanti bit di stop usa il protocollo di comunicazione.

Di regola, anche per la nostra BBS, è uno solo.

QUANTO COSTA IL TUO COMMODORE

Amiga 2000 - L. 2.715.000

Microprocessore Motorola MC68000 - Clock 7.16MHz - Kickstart ROM - Memoria RAM: 1 MByte - 3 chip custom per DMA, Video, Audio, I/O - 5 Slot di Espansione Amiga Bus 100 pin AutoconfigTM - 1 Slot di Espansione 86 pin per Schede Coprocessore - 2 Slot di Espansione compatibili AT/XT - 2 Slot di Espansione compatibili XT - 2 Slot di Espansione Video - 1 Floppy Disk Drive da 3 1/2", 880 KBytes - Porta seriale RS232C - Sistema Operativo single-user, multitasking AmigaDOS - Compatibilità MS-DOS XT/AT disponibile con schede interne Janus (A2088 - A2286) - Monitor escluso

Amiga 500 - L. 995.000

Microprocessore Motorola MC68000 - Clock 7.16 MHz - Kickstart ROM - Memoria RAM: 512 KBytes - 3 Chip custom per DMA, Video, Audio, I/O - 1 Floppy Disk Driver da 3 1/2", 880 KBytes - Porta seriale RS232C - Porta parallela Centronics

Videomaster 2995 - L. 1.200.000

Desk Top Video - Sistema per elaborazini video semiprofessionale composto da genlock, digitalizzatore e alloggiamento per 3 drive A2010 - Ingressi videocomposito (2), RGB - Uscite Videocomposito, RF, RGB + sync -

Floppy Disk Driver A 1010 - L. 335.000

Floppy Disk Driver - Drive esterno da 3 1/2" - Capacità 880 KBytes - Collegabile a tutti i modelli della linea Amiga, alla scheda A2088 e al PC1

Floppy Disk Drive A 2010 - L. 280.000

Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" - Capacità 880 KBytes - Collegabile ad Amiga 2000

Hard Disk A 590 - L. 1.750.000

Hard Disk+Controller+RAM - Scheda Controller - Hard Disk da 3 1/2" 20 MBytes - 2 MBytes "fast" RAM - Collegabile all'Amiga 500

Scheda Janus A 2088 + A 2020 - L. 1.050.000

Scheda Janus XT+Floppy Disk Drive da 5 1/4", 360 KBytes - Scheda Bridgeboard per compatibilità MS-DOS (XT) in Amiga 2000 - Microprocessore Intel 8088 - Coprocessore matematico opzionale Intel 8087

A2286+A2020 - L. 1.985.000

Scheda Janus AT+Floppy Disk Drive da 5 1/4", 1.2 MBytes - Scheda Bridgeboard per compatibilità MS-DOS (AT) in Amiga 2000 - Microprocessore Intel 80287 - Clock 8 MHz - RAM: 1 MBytes on-board - Floppy Disk Controller on-board - Floppy Disk Driver disegnato per l'installazione all'interno dell'Amiga 2000 -

Scheda A2620 - L. 2.700.000

Scheda Processore Alternativo 32 bit - Scheda per 68020 e Unix - Microprocessore Motorola MC68020 - Coprocessore matematico Motorola MC68881 (opzionale MC68882)

Scheda A Unix - L. 3.250.000

Sistema Operativo AT&T Unix System V Release 3 - Per Amiga 2000 con scheda A2620 e Hard Disk 100 MBytes

Hard Disk A2092+PC5060 - L. 1.020.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+2092 - L. 1.240.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+A2094 - L. 1.900.000

Stesse caratteristiche del kit A2092 ma con disco da 40 MBytes

Espansione di memoria A2058 - L. 1.149.000

Espansione di memoria - Scheda di espansione per Amiga 2000 - Fornita con 2 MBytes "fast" RAM, espandibile a 4 o 8 MBytes

Scheda Video A2060 - L. 165.000

Modulatore video - Scheda modulatore video interna per Amiga 2000 - Uscite colore e monocromatica - Si inserisce nello slot video dell'Amiga 2000

Genlock Card A2301 - L. 420.000

Genlock - Scheda Genlock semiprofessionale per Amiga 2000 - Permette di miscelare immagini provenienti da una sorgente esterna con immagini provenienti dal computer

Professional Video Adapter Card A2351 - L. 1.500.000

Professional Video Adapter - Scheda Video Professionale per Amiga 2000 (B) - Genlock qualità Broadcast - Frame Grabber - Digitalizzatore - Include software di controllo per la gestione interattiva
(Disponibile da maggio '89)

A501 - L. 300.000

Espansione di memoria - Cartuccia di espansione di memoria da 512 KBytes per A500

A520 - L. 45.000

Modulatore RF - Modulatore esterno A500 - Permette di connettere qualsiasi televisore B/N o colori ad Amiga 500

A Scart - L. 27.000
Cavo di collegamento A500/A2000 con connettore per televisione SCART

Monitor a colori 1084 - L. 595.000
Monitor a colori ad alta risoluzione - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor a colori 2080 - L. 770.000
Monitor a colori ad alta risoluzione e lunga persistenza - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Frequenza di raster 50 Hz - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor Monocromatico A2024 - L. 1.235.000
Monitor monocromatico a fosfori "bianco-carta" - Turbo 14" antiriflesso - (Disponibile da marzo '89)

PC60/40 - L. 7.812.000
Microprocessore Intel 80386 - Coprocessore matematico opzionale Intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 funzioni - Sistema Operativo MS-DOS 3 2.1 - Interprete GW-Basic

PC60/40C - L. 8.127.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 60/80 - L. 10.450.000
Microprocessore Intel 80386 - Coprocessore opzionale Intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Memoria RAM: 2.5 MBytes - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Floppy Disk Drive opzionale da 3 1/2", 1.44 MBytes - 1 Hard Disk da 80 MBytes - 2 Porte parallele Centronics - Mouse video EGA (compatibile MDA - Hercules - CGA). Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Ambiente Operativo Microsoft Windows/386 - Interprete GW-Basic

PC60/80C - L. 10.700.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC40/20 - L. 4.100.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/20C - L. 4.350.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 40/40 - L. 5.285.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/40C - L. 5.535.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

1352 - L. 78.000
Mouse - Collegabile con Microsoft Bus Mouse - Collegabile direttamente a PC1, PC10/20 - III, PC40 - III

PC910 - L. 355.000
Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" per PC10/20-I-II-III - Capacità 360 o 720 KBytes selezionabile tramite "config. sys" - Corredo di telaio di supporto per l'installazione in un alloggiamento per un drive da 5 1/4" - Interfaccia identica ai modelli da 5 1/4"

PC1 - L. 995.000
Microprocessore Intel 8088 - 1 Floppy Disk Drive da 5 1/4" - Porta seriale RS232C - Porta parallela Centronics - - Monitor monocromatico 12" - Tastiera 84 tasti - Sistema Operativo MS-DOS 3.2 - Interprete GW-Basic

PCEXP1 - L. 640.000
PC Expansion Box - Box esterno di espansione per PC 1 - Alimentatore aggiuntivo incluso - Contiene 3 Slot di Espansione compatibili Ibm XT - Alloggiamento per Hard Disk da 5 1/4" - Si posiziona sotto il corpo del PC1 e viene collegato tramite degli appositi connettori

PC10-III - L. 1.360.000
Microprocessore Intel 8088 Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - Memoria RAM: 640 KBytes - 2 Floppy Disk Drive da 5 1/4", 360 KBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC10-IIIC - L. 1.675.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC20-III - L. 2.095.000
Microprocessore Intel 8088 - Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - 1/4", 360 KBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse) - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC20-IIIC - L. 2.410.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

Nuovo C64 - L. 325.000
Nuovo Personal Computer CPU 64 KBytes RAM - Vastissima biblioteca software disponibile - Porta seriale Commodore - Porta registratore per cassette - Porta parallela programmabile -

C128D - L. 895.000
Personal Computer CPU 128 KBytes RAM espandibile a 512 KBytes - ROM 48 KBytes - Basic 7.0 - Tastiera separata - Funzionante in modo 128,64 o CP/M 3.0 - Include floppy disk drive da 340 KBytes

Floppy Disk Drive 1541 II - L. 365.000
Floppy Disk Drive - Floppy Disk Drive da 5 1/4" singola faccia - Capacità 170 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

Floppy Disk Dirve 1581 - L. 420.000
Floppy Disk Drive da 3 1/2" doppia faccia - Capacità 800 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

1530 - L. 55.000
Registratore a cassette per C64, C128, C128D

Accessori per C64 - 128D
1700 - Espansione di memoria - Cartuccia di espansione di memoria a 128 KBytes per C128 **- L. 170.000**

1750 - Espansione di memoria - Cartuccia di espansione di memoria 512 KBytes per C128 - **L. 245.000**

1764 - Espansione di memoria - Cartuccia di espansione di memoria a 256 KBytes per C64 - Fornita di alimentatore surdimensionato - **L. 198.000**

16499 - Adattatore Telematico Omologato - Collegabile al C64 - Permette il collegamento a Videotel, P.G.E. e banche dati **L. 149.000**

1399 - Joystick - Joystick a microswitch con autofire - **L. 29.000**

1351 - Mouse - Mouse per C64, C128, C128D - **L. 72.000**

Monitor Monocromatico 1402 - L. 280.000
Monitor monocromatico a fosfori "bianco-carta" - Turbo 12" antiriflesso - Ingresso TTL - Compatibile con tutta la gamma PC

Monitor Monocromatico 1404 - L. 365.000
Monitor monocromatico a fosfori ambra - Turbo 14" antiriflesso a schermo piatto - Ingresso TTL - Compatibile con tutta la gamma PC - Base orientabile

Monitor Monocromatico 1450 - L. 470.000
Monitor monocromatico BI-SYNC a fosfori "bianco-carta" - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Monitor a colori 1802 - L. 445.000
Monitor a colori - Turbo 14" - Collegabile a C64, C128, C128D

Monitor monocromatico 1900 - L. 199.000
Monitor monocromatico a fosfori verdi - Turbo 12" antiriflesso - Ingresso videocomposito - Compatibile con tutta la gamma Commodore

Monitor a colori 1950 - L. 1.280.000
Monitor a colori BI-SYNC alta risoluzione - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Stampante MPS 1230 - L. 465.000
Stampante a matrice di punti - Testina a 9 aghi - 120 cps - Bidirezionale - 80 colonne - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

MPS 1230R - L. 19.000
Nastro per stampante

Stampante MPS 1500C - L. 495.000
Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia parallela Centronics - Compatibile con la gamma Amiga e PC

MPS1500R - L. 37.000
Nastro a colori per stampante

MPS1500R - L.37.000
Nastro a colori per stampante

Stampante MPS 1550C - L. 575.000
Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

I COMMODORE POINT

LOMBARDIA

Milano
• AL RISPARMIO - V.LE MONZA 204
• BCS - VIA MONTEGANI 11
• BRAHA A. - VIA PIER CAPPONI 5
• E.D.S. - C.SO PORTA TICINESE 4
• FAREF - VIA A. VOLTA 21
• FLOPPERIA - V.LE MONTENERO 31
• GBC - VIA CANTONI 7 - VIA PETRELLA 6
• GIGLIONI - V.LE LUIGI STURZO 45
• L'UFFICIO 2000 - VIA RIPAMONTI 213
• LOGITEK - VIA GOLGI 60
• LU - MEN - VIA SANTA MONICA 3
• MARCUCCI - VIA F.LLI BRONZETTI 37
• MELCHIONI - VIA P. COLLETTA 37
• MESSAGGERIE MUSICALI - GALLERIA DEL CORSO 2
• NEWEL - VIA MAC MAHON 75
• PANCOMMERZ ITALIA - VIA PADOVA 1
• SUPERGAMES - VIA VITRUVIO 38
• 68000 E DINTORNI - VIA WASHINGTON 91

Provincia di Milano
• GINO FERRARI CENTRO HI-FI - VIA MADRE CABRINI 44 - S. ANG. LODIGIANO
• F.LLI GALIMBERTI - VIA NAZIONALE DEI GIOVI 28/36 - BARLASSINA
• TECNOLUX - VIA PIETRO NENNI 5 - BERNATE TICINO
• OGGIONI & C. - VIA DANTE CESANA 27 - CARATE BRIANZA
• AL RISPARMIO - VIA U. GIORDANO 57 - CINISELLO BALSAMO
• GBC - V.LE MATTEOTTI 66 - CINISELLO BALSAMO
• CASA DELLA MUSICA - VIA INDIPENDENZA 21 - COLOGNO MONZESE
• PENATI - VIA VERDI 28/30 - CORBETTA
• EPM SYSTEM - V.LE ITALIA 12 - CORSICO
• P.G. OSTELLARI - VIA MILANO 300 - DESIO
• CENTRO COMPUTER PANDOLFI - VIA CORRIDONI 18 - LEGNANO
• COMPUTEAM - VIA VECELLIO 41 - LISSONE
• M.B.M. - C.SO ROMA 112 - LODI
• L'AMICO DEL COMPUTER - VIA CASTELLINI 27 - MELEGNANO
• BIT 84 - VIA ITALIA 4 - MONZA
• IL CURSORE - VIA CAMPO DEI FIORI 35 - NOVATE MIL.
• I.C.O. - VIA DEI TIGLI 14 - OPERA
• R & C ELGRA - VIA SAN MARTINO 13 - PALAZZOLO MIL.
• ESSEGIEMME SISTEMI SAS - VIA DE AMICIS 24 - RHO
• TECNO - CENTRO - VIA BARACCA 2 - SEREGNO
• NIWA HARD&SOFT - VIA B. BUOZZI 94 - SESTO SAN GIOV.
• COMPUTER SHOP - VIA CONFALONIERI 35 - VILLASANTA
• ACTE - VIA B. CREMIGNANI 13 - VIMERCATE
• IL COMPUTER SERVICE SHOP - VIA PADANA SUPERIORE 197 - VIMODRONE

Bergamo
•D.R.B. - VIA BORGO PALAZZO 65
• TINTORI ENRICO &C. - VIA BROSETA 1
• VIDEO IMMAGINE - VIA CARDUCCI c/o CITTA' DI MERCATO

Provincia di Bergamo
BERTULEZZI GIOVANNI - VIA FANTONI 48 - ALZANO LOMBARDO
• COMPUTER SHOP - VIA VITTORIO VENETO 9 - CAPRIATE SAN GERVASIO
• B.M.R. - VIA BUTTARO 4/T - DALMINE
• MEGABYTE 2 - VIA ROMA 61/A - GRUMELLO
• OTTICO OPTOMETRISTA ROVETTA - P.ZZA GARIBALDI 6 - LOVERE
• COMPUTER POINT - VIA LANTIERI 52 - SARNICO
• A.B. INFORMATICA - STRADA STATALE CREMASCA 66 - URGNANO

Brescia
• MASTER INFORMATICA - VIA F.LLI UGONI 10/B

PROVINCIA DI BRESCIA
• MISTER BIT - VIA MAZZINI 70 - BRENO
• CAVALLI PIETRO - VIA 10 GIORNATE 14 BIS - CASTREZZATO
• VIETTI GIUSEPPE - VIA MILANO 1/B - CHIARI
• MEGABYTE - P.ZZA MALUEZZI 14 - DESENZANO DEL GARDA
• BARESI RINO &C. - VIA XX SETTEMBRE 7 - GHEDI
• INFO CAM - VIA PROVINCIALE 3 - GRATACASOLO
• "PAC-LAND" di GARDONI - CENTRO COM.le - LA CASA DI MARGHERITA D'ESTE - VIA GIORGIONI 21

Como
• IL COMPUTER - VIA INDIPENDENZA 90
• 2M ELETTRONICA - VIA SACCO 3

Provincia di Como
• ELTRON - VIA IV NOVEMBRE 1 - BARZANO'
• DATA FOUND - VIA A. VOLTA 4 -ERBA
• CIMA ELETTRONICA - VIA L. DA VINCI 7 - LECCO
• FUMAGALLI - VIA CAIROLI 48 -LECCO
• RIGHI ELETTRONICA - VIA G. LEOPARDI 26 - OLGIATE COMASCO

Cremona
• MONDO COMPUTER - VIA GIUSEPPINA 11/B
• PRISMA - VIA BUOSO DA DOVARA 8
• TELCO - P.ZZA MARCONI 2/A

Provincia di Cremona
• ELCOM - VIA IV NOVEMBRE 56/58 - CREMA
• EUROELETTRONICA - VIA XX SETTEMBRE 92/A - CREMA

Mantova
• COMPUTER CANOSSA - GAL. FERRI 7
• 32 BIT - VIA C. BATTISTI 14
• ELET. di BASSO - V.LE RISORGIMENTO 69

Provincia di Mantova
• CLICK - ON COMPUTER - S.S. GOITESE 168 - GOITO

Pavia
• POLIWARE - C.SO C. ALBERTO 76
• SENNA GIANFRANCO - VIA CALCHI 5

Provincia di Pavia
• A. FERRARI - C.SO CAVOUR 57 - MORTARA
• LOGICA MAINT - V.LE M.TE GRAPPA 32 - VIGEVANO
• M. VISENTIN - C.SO V. EMANUELE 76 - VIGEVANO

Sondrio
• CIPOLLA MAURO - VIA TREMOGGE 25

Provincia di Sondrio
• FOTONOVA - VIA VALERIANA 1 - S.PIETRO DI BERBENNO

Varese
• ELLE - EFFE - VIA GOLDONI 35
• IL C.TRO ELET. - VIA MORAZZONE 2
• SUPERGAMES - VIA CARROBBIO 13

Provincia di Varese
• BUSTO BIT - VIA GAVINANA 17-BUSTO A.
• MASTER PIX - VIA S.MICHELE 3 - BUSTO A.
• PUNTO UFFICIO - VIA R.SANZIO 8 - GALLARATE
• GRANDI MAGAZZINI BOSSI - VIA CLERICI 196 - GERENZANO
• J.A.C. - C.so MATTEOTTI 38 - SESTO C.

PIEMONTE

Alessandria
• BIT MICRO - VIA MAZZINI 102
• SERV. INFOR. - VIA ALESSANDRO III 47

Provincia di Alessandria
• SONY ITALIANA - VIA G. MANARA 7 - CASALE MONFERRATO
• SGE ELETTRONICA - VIA BANDELLO 19 - TORTONA
• COMPUTER TEMPLE - VIA F. CAVALLOTTI 13 - VALENZA

Asti
• ASTI GAMES - C.SO ALFIERI 26
• RECORD - C.SO ALFIERI 166/3 (Galleria Argenta)

Cuneo
• ROSSI COMPUTERS - C.SO NIZZA 42

Provincia di Cuneo
• PUNTO BIT - C.SO LANGHE 26/C - ALBA
• BOSETTI - VIA ROMA 149 - FOSSANO
• COMPUTERLAND - VIA MAZZINI 30/32 - SALUZZO

Novara
• PROGRAMMA 3 - V.LE BUONARROTI 8
• PUNTO VIDEO - C.so RISORGIMENTO 39/B

Provincia di Novara
• COMPUTER - VIA MONTE ZEDA 4 -ARONA
• ALL COMPUTER - C.SO GARIBALDI 106 - BORGOMANERO
• S.P.A. - C.SO DISSEGNA 21/BIS - DOMODOSSOLA
• ELLIOTT COMPUTER SHOP - VIA DON MINZONI 32 - INTRA
• TRISCONI VALERIA - VIA MAZZINI 90 - OMEGNA

Torino
• ABA ELETTRONICA - VIA C. FOSSATI 5/P
• ALEX COMPUTER E GIOCHI - C.SO FRANCIA 333/4
• COMPUTER HOME - VIA SAN DONATO 46/D
• COMPUTING NEW - VIA M. POLO 40/E
• C.D.M. ELETTR. - VIA MAROCHETTI 17
• DE BUG - C.SO V. EMANUELE II 22
• DESME UNIVERSAL - VIA S.SECONDO 95
• FDS ALTERIO - VIA BORGARO 86/D
• IL COMPUTER - VIA N. FABRIZI 126
• MICRONTEL - C.SO D. degli ABRUZZI 28
• PLAY GAMES SHOP - VIA C. ALBERTO 39/E
• RADIO TV MIRAFIORI - C.SO UNIONE SOVIETICA 381
• SMT ELETTRONICA - VIA BIBIANA 83/bis

Provincia di Torino
• PAUL E CHICO VIDEOSOUND - VIA V.EMANUELE 52 - CHIERI
• BIT INFORMATICA - VIA V. EMANUELE 154 - CIRIE'
• HI - FI CLUB - C.SO FRANCIA 92C - COLLEGNO
• MISTER PERSONAL - VIA CATTANEO 52 - FAVRIA
• I.C.S. - VIA TORINO 73 - IVREA
• DAG - VIA I MAGGIO 40 - LUSERNA S. GIOVANNI
• EUREX - C.SO INDIPENDENZA 5 - RIVAROLO CANAVESE
• DIAM INFORMATICA - C.SO FRANCIA 146 bis - RIVOLI
• FULLINFORMATICA - VIA V.VENETO 25 - RIVOLI
• GAMMA COMPUTER - VIA CAVOUR 3A-3B - SET.TORINESE

Vercelli
•ELETTROGAMMA - C.SO BORMIDA 27 ang. V.Montanara
• ELETTRONICA - STRADA TORINO 15

Provincia di Vercelli
• C.S.I. TEOREMA - VIA LOSANA 9 - BIELLA
• SIGEST - VIA BERTODANO 8 - BIELLA
• REMONDINO FRANCO - VIA ROMA 5 - BORGOSESIA
• FOTOSTUDIO TREVISAN - VIA XXV APRILE 24/B - COSSATO
• STUDIO FOTOGRAFICO IMARISIO - P.ZZA M. LIBERTA' 7 - TRINO

VENETO

Belluno
• UP TO DATE - VIA V. VENETO 43

Provincia di Belluno
• GUERRA COMPUTERS - V.LE MAZZINI 10/A - FELTRE

Padova
• BIT SHOP - VIA CAIROLI 11
• COMPUMANIA - VIA T. CAMPOSANPIERO 37
• D.P.R. DE PRATO R. - V.LO LOMBARDO 4
• G.F. MARCATO - VIA MADONNA DELLA SALUTE 51/53
• SARTO COMPUTER - VIA ARMISTIZIO 79

Provincia di Padova
• COMPUTER SERVICE - BORGO TREVISO 150 - CITTADELLA

Treviso
• BIT 2000 - VIA BRANDOLINI D'ADDA 14
• GUERRA EGIDIO & C. - V.LE CAIROLI 95

Provincia di Treviso
• DE MARIN COMPUTERS - VIA MATTEOTTI 142 - CONEGLIANO
• SIDESTREET - VIA SALVO D'ACQUISTO 8 - MONTEBELLUNA
• FALCON ELETTROAUDIOVIDEO - VIA TERRAGGIO 116 - PREGANZIOL

Venezia
• GUERRA EGIDIO & C. - VIA BISSUOLA 20/A - MESTRE
• TELERADIO FUGA - SAN MARCO 3457

Provincia di Venezia
• GUERRA EGIDIO & C. - VIA VIZZOTTO 29 - SAN DONA' DI PIAVE
• REBEL - VIA F. CRISPI 10 - SAN DONA' DI PIAVE

Verona
• CASA DELLA RADIO - VIA CAIROLI 10
• TELESAT - VIA VASCO DE GAMA 8

• Provincia di Verona
• UBER - CP 0363(RAG.SOC. DERTA) - VIA MASCAGNI 31 -CASTE' D'AZZANO
• FERRARIN - VIA DEI MASSARI 10 - LEGNAGO
• COMPUTERS CENTER - VIA CANTORE 26 - VILLAFRANCA

Vicenza
• ELET. BISELLO - V.LE TRIESTE 427/429
• SCALCHI MARKET - VIA C.A' BALBI 139

Provincia di Vicenza
• SCHIAVOTTO - VIA ZANELLA 21 - CAVAZZALE
• GUERRA E. & C. - V.LE DELLE INDUSTRIE - MONTECCHIO MAGGIORE

FRIULI VENEZIA GIULIA

Gorizia
• E.CO. ELETTRONICA - VIA F.LLI COSSAR 23

Trieste
• AVANZO GIACOMO - P.ZZA CAVANA 7
• COMPUTER SHOP - VIA P. RETI 6
• COMPUTIGI - VIA XX SETTEMBRE 51
• CTI - VIA PASCOLI 4

Udine
• MOFERT 2 - VIA LEOPARDI 21
• R.T. SISTEM UDINE - VIA L. DA VINCI 99

Provincia di Udine
• IL PUNTO ELETTRONICO - VIA VENDRAMIN 184 - LATISANA
• IDRENO MATTIUSSI &C. - VIA LICINIANA 58 - MARTIGNACCO

TRENTINO ALTO ADIGE

Bolzano
• COMPUTER POINT - VIA ROMA 82/A
• MATTEUCCI PRESTIGE - VIA MUSEO 54

Provincia di Bolzano
• RADIO MAIR-ELECTRO - VIA CENTRALE 70 - BRUNICO
• ELECTRO RADIO HENDRICH - VIA DELLE CORSE 106 - MERANO
• ERICH KONTSCHIEDER - PORTICI 313 - MERANO
• ELECTRO TAPPEINER - P.ZZA PRINCIPALE 90 - SILANDRO

Trent0
• CRONST - VIA G. GALILEI 25

Provincia di Trento

• AL RISPARMIO - C.SO VERONA 138 - ROVERETO

LIGURIA
Genova
• ABM COMPUTER - P.ZZA DE FERRARI 24 rosso
• CAPRIOTTI G. - IA MAMIANI 4r - SAMPIERDARENA
• C.tro ELET. - VIA CHIARAVAGNA 10 R - VIA SESTRI 69R
• COM.le SOTTORIPA - VIA SOTTORIPA 115/117
• FOTOMONDIAL - VIA DEL CAMPO 3-5-9-11-13 r
• LA NASCENTE - VIA SAN LUCA 4/1
• PLAY TIME - VIA GRAMSCI 3/5/7 rosso
• RAPPR-EL - VIA BORGORATTI 23 R
Imperia
• CASTELLINO - VIA BELGRANO 44
Provincia di Imperia
• CENTRO HI-FI VIDEO - VIA DELLA REPUBBLICA 38 -SANREMO
• CASTELLINO - VIA GENOVA 48 - VENTIMIGLIA
La Spezia
• I.L. ELETTRONICA - VIA V. VENETO 123
Provincia di La Spezia
• I.L. ELETTRONICA - VIA AURELIA 299 - FORNOLA DI VEZZANO
Savona
• CASTELLINO - C.SO TARDY E BENECH 101
Provincia di Savona
• CELESIA ENZA - VIA GARIBALDI 144 - LOANO

EMILIA
Bologna
• EUROELETTRICA - VIA RANZANI 13/2
• MINNELLA ALTA FEDELTA' - VIA MAZZINI 146/2
• MORINI & FEDERICI - VIA MARCONI 28/C
• STERLINO - VIA MURRI 73/75
Provincia di Bologna
• S.C. COMPUTERS - VIA E. FERMI 4 - CASTEL SAN PIETRO
• S.P.E. INFORMATICA - VIA DI MEZZO PONENTE 385 - CREVALCORE
• ARCHIMEDE SISTEMI - VIA EMILIA 124 - S. LAZZARO DI SAVENA
Modena
• CO - EL - VIA CESARI 7
• ORSA MAGGIORE - P.ZZA MATTEOTTI 20
• VIDEO VAL WILLY COMPUTERS - VIA CANALETTO 223
Provincia di Modena
• NEW MEDIA SYSTEM - VIA ROMA 281 - SOLIERA
Parma
• BABARELLI G. - VIA B. PARENTE 14/A/B
Provincia di Parma
• PONGOLINI - VIA CAVOUR 32 - FIDENZA
Piacenza
• COMPUTER LINE - VIA G. CARDUCCI 4
• DELTA COMPUTER - VIA M. DELLA RESISTENZA 15/G

TEGGIO EMILIA
• COMPUTERLINE - VIA SAN ROCCO 10/C
• POOL SHOP - VIA EMILIA S. STEFANO 9/C
Provincia di Reggio Emilia
• MACCHIONI - VIA STATALE 467 - CASALGRANDE
ROMAGNA
Ferrara
• BUSINESS POINT - VIA CARLO MAYER 85
Forlì
• COMPUTER VIDEO CENTER - VIA CAMPO DI MARTE 122
Provincia di Forlì
• TOP BIT - VIA VENETO 12 - FORLIMPOPOLI
• COMPUTER HOUSE - V.LE TRIPOLI 193/D - RIMINI
• EASY COMPUTER - VIA LAGOMAGGIO 50 - RIMINI

REPUBBLICA S. MARINO
Ravenna
• COMPUTER HOUSE - VIA TRIESTE 134
Provincia di Ravenna
• ARGNANI - P.ZZA DELLA LIBERTA 5/A - FAENZA
• ELECTRON INFORMATICA - VIA F.LLI CORTESI 17 - LUGO
• P.L.Z. INFORMATICA - P.ZZA SERCOGNANI 6 - FAENZA

TOSCANA
Arezzo
• DELTA SYSTEM - VIA PIAVE 13
Firenze
• ATEMA - VIA BENEDETTO MARCELLO 1a-1b
• ELETTRONICA CENTOSTELLE - VIA CENTO STELLE 5/a-b
• HELP COMPUTER - VIA DEGLI ARTISTI 15-A
• TELEINFORMATICA TOSCANA -VIA BRONZINO 36
Provincia di Firenze
• WAR GAMES - VIA R. SANZIO 126/A - EMPOLI
• NEW EVM COMPUTER - VIA DEGLI INNOCENTI 2 - FIGLINE VALDARNO
• C.tro INFOR. - VIA ZNOJMO 41 - PONTASSIEVE
• COSCI F.LLI - VIA ROMA 26 - PRATO
• BARBAGLI C. ELET. - VIA F. BONI 80 - PRATO
Grosseto
• COMPUTER SERVICE - VIA DELL UNIONE 7
Livorno
• ETA BETA - VIA SAN FRANCESCO 30
• FUTURA 2 - VIA CAMBINI 19
Provincia di Livorno
• PUNTO ROSSO - VIA BARONTINI 28 - PIOMBINO
Provincia di Lucca
• IL COMPUTER - V.LE COLOMBO 216 - LIDO DI CAMAIORE
• SANTI VITTORIO - VIA ROMA 23 - S. ROMANO GARFAGNANA
• TOP GAMES - VIA S. ANDREA 122 - VIAREGGIO
Massa
• EURO COMPUTER - P.ZZA G. BERTAGNINI 4
Carrara
• RADIO LUCONI - VIA ROMA 24/B
Pisa
• ELECTRONIC SERVICE - VIA DELLA VECCHIA TRANVIA 10
• PUCCINI S.- CP 1199 (RAG.SOC. MAREX) - VIA C.CAMMEO 64
• TONY HI-FI - VIA CARDUCCI
Provincia di Pisa
• M.C. INFORMATICA - VIA DEL CHIESINO 4 - PONTEDERA (PI)
Pistoia
• ELECTRONIC SHOP - VIA DEGLI SCALZI 3
Provincia di Pistoia
• ZANNI &C. - C.SO ROMA 45 - MONTECATINI T.

Siena
• R. BROGI - P.ZZA GRAMSCI 28
• VIDEO MOVIE - VIA GARIBALDI 17
Provincia di Siena
• ELETTRONICA di BIFOLCHI - VIA DI GRACCIANO NEL CORSO 111 - MONTEPULCIANO

LAZIO
• CENTRO INF. - D.R.R. srl - TEL. 06-5565672

UMBRIA
Perugia
• MIGLIORATI - VIA S. ERCOLANO 3-10
Provincia di Perugia
• COMPUTER STUDIO'S - VIA IV NOVEMBRE 18/A - BASTIA UMBRA
• WARE - VIA DEI CASCERI 31 - CITTA'DI CASTELLO
Terni
• CGS SOFTWARE HOUSE - VIA DONIZETTI 71/A

BASILICATA
Matera
• G. GAUDIANO ELECTRONICS - VIA ROMA ang. XX SETTEMBRE 1

PUGLIA
Bari
• ARTEL - VIA GUIDO D'ORSO 9
• COMPUTER'S ARTS - V.LE MEUCCI 12/B
• PAULICELLI S. & F. - VIA FANELLI 231/C
Provincia di Bari
• F. FAGGELLA - C.SO GARIBALDI 15 - BARLETTA
• G.FAGGELLA - P.ZZA D'ARAGONA 62A - BARLETTA
• LONUZZO G. - VIA NIZZA 21 - CASTELLANA
• TECNOUFF. - VIA RICASOLI 54 - MONOPOLI
• TANGORRA N. - C.SO V.EMANUELE 130/B - TRIGGIANO
Brindisi
• MARANGI E NICCOLI - VIA PROV. SAN VITO 165
Provincia di Brindisi
• MILONE G. - VIA S.F. D'ASSISI 219 - FRANCAVILLA FONTANA
Foggia
• BOTTICELLI G. - VIA SAV POLLICE 2
• E.C.I. COMPUTER - VIA ISONZO 28
• LA TORRE - V.LE MICHELANGELO 185
Provincia di Foggia
• IL DISCOBOLO - VIA T. SOLIS 15 - SAN SEVERO
Lecce
• BIT - VIA 95 REGG.NTO FANTERIA 87/89
Provincia di Lecce
• TECNO UFFICIO - P.ZZA GIOVANNI XXIII 10 - GALLIPOLI
• CEDOK INFORMATICA - VIA UMBERTO I 116 - TRICASE
Taranto
• ELETTROJOLLY C.tro - VIA DE CESARE 13
• TEA - TEC. ELET. AV. - VIA R. ELENA 101

CAMPANIA
Provincia di Avellino
• FLIP FLOP - VIA APPIA 68 - ATRIPALDA
Benevento
• E.CO. INF. - VIA PEPICELLI 21/25
Caserta
• ENTRY POINT - VIA COLOMBO 31
• O.P.C. - VIA G. M. BOSCO 24
Provincia di Caserta
• M.P. COMPUTER - VIA NAPOLI 30 - MADDALONI
• DAMIANO - C.SO V. EMANUELE 23 -ORTA DI ATELLA
• FUSCO B. - VIA NAPOLI 24 - VAIRANO PATERNORA (FRAZ. VAIRANO SCALO)
• LINEA CONTABILE - VIA OSPEDALE 72/76 - SESSA A. (CE)
Napoli
• BABY TOYS - VIA CISTERNA DELL'OLIO 5/BIS
• CASA MUSICALE RUGGIERO - P.ZZA GARIBALDI 74 (INT. STAZ. F.F. S.S.)
• C.tro ELET. CAMPANO - VIA EPOMEO 121
• CI.AN - GALLERIA VANVITELLI 32
• CINE NAPOLI - VIA S. LUCIA 93/95
• DARVIN - CALATA SAN MARCO 26
• GIANCAR 2 - P.ZZA GARIBALDI 37
• ODORINO - L.GO LALA 22 A-B
• R 2 - VIA F. CILEA 285
• SAGMAR - VIA S. LUCIA 140
• TOP VIDEO - TOP COMPUTER - VIA S. ANNA DEI LOMBARDI 12
• VIDEOFOTOMARKET - VIA S. BRIGIDA 19
Provincia di Napoli
• ELECTRONIC DAY - VIA DELLE PUGLIE 17 - CASORIA
• TUFANO - S.S. SANNITICA 87 KM 7 - CASORIA
• SOF SUD - V.LE EUROPA 59 - CASTEL/MARE DI STABIA
• ELETTRONICA 2000 - C.SO DURANTE 40 - FRATTAMAGGIORE
• SPADARO - VIA ROMANI 93 - MADONNA DELL'ARCO
• GATEWAY - VIA NAPOLI 68 - MUGNANO
• VISPINI & DI VUOLO - VIA A.ROSSI 4 - POMPEI
• SPY CASH & CARRY - P.ZZA ARENELLA 6/A - NAPOLI
• NUOVA INFORMATICA SHOP - VIA LIBERTA' 185/191 - PORTICI
• BASIC COMPUTER - C.SO GARIBALDI 34 - POZZUOLI
• V.C. - C.SO SECONDIGLIANO 562/B - SECONDIGLIANO
• F. ELETTRONICA - VIA SARNO 102 - STRIANO
• TECNO - VIA V. VENETO 48 - TORRE DEL GRECO
Salerno
• COMPUMARKET - VIA BELVEDERE 35
• COMPUTER MARKET - C.SO VITTORIO EMANUELE 23
Provincia di Salerno
• KING COMPUTER - VIA OLEVANO 56 - BATTIPAGLIA
• DIMER POINT - V.LE AMENDOLA 36 - EBOLI
• IACUZIO F. - VIA MUNICIPIO 14 - MERCATO SAN SEVERINO
• COMPUTER SERVICE - VIA L.DA VINCI 81 - SCAFATI

CALABRIA
Catanzaro
• C. & G. COMPUTER - VIA F. ACRI 28
• PAONE S. & F. - VIA F. ACRI 93/99
Provincia di Catanzaro
• COMPUTER HOUSE - VIA BOLOGNA (L.GO OSPEDALE) - CROTONE
• RIOLO F.LLI - VIA VENEZIA 1/7 - CROTONE
• ING. FUSTO S. - C.SO NICOTERA 99 - LAMEZIA TERME
Cosenza
• MAISON DE L'INFORMATIQUE - VIA PASQUALE ROSSI 34/C
• SIRANGELO COMP. - VIA N. PARISIO 25
Provincia di Cosenza
• HI-FI ALFANO G. - VIA BALDACCHINI 109 - AMANTIA
• ELIGIO ANNICCHIARICO &C. - VIA ROMA 21 - CASTROVILLARI
• ALFA COMPUTER - VIA NAZIONALE 341/A - CORIGLIANO SCALO
REGGIO CALABRIA
• CONTROL SYSTEM - VIA S.F. DA PAOLA 49 D
• SYSTEM HOU. - VIA FIUME ang. PALESTINO 1
Provincia di Reggio Calabria
• COMPUTER SHOP - V.LE MATTEOTTI 36/38 - LOCRI
• PICIEFFE - C.SO F. S. ALESSIO 19 - TAURIANOVA
SICILIA
• CENTRO INF. - ITALSOFT SRL - TEL. 0935-696090

Leggo VR perché mi dà la rotta

IMMAGINE MARSILII